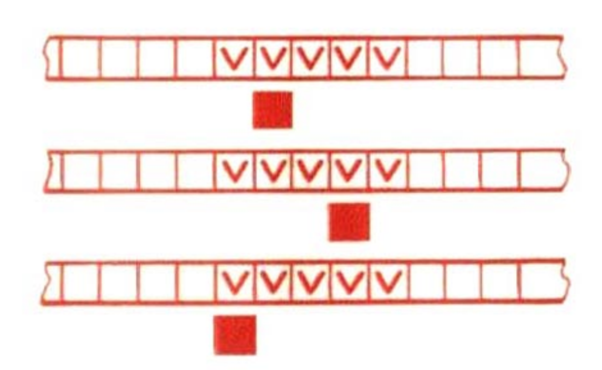
Lecciones populares de matemáticas

MÁQUINA DE POST

V. A. Uspenski



Editorial MIR


Moscú


ИЗДАТЕЛЬСТВО
МИР

ПОПУЛЯРНЫЕ ЛЕКЦИИ ПО МАТЕМАТИКЕ

В. А. УСПЕНСКИЙ

МАШИНА ПОСТА

ИЗДАТЕЛЬСТВО «НАУКА»

LECCIONES POPULARES DE MATEMÁTICAS

V. A. USPENSKI

MÁQUINA DE POST

TRADUCIDO DEL RUSO POR
STANISLAV N. BELOUSOV

EDITORIAL MIR
MOSCÚ

IMPRESO EN LA URSS

На испанском языке

INDICE

PREFACIO

Este libro se destina, ante todo, a los escolares. El contenido de los dos primeros capítulos es comprensible, incluso para los alumnos de los grados primarios. En el libro se expone cierta máquina computadora «de juguete» («abstracta», en sentido científico), la llamada máquina de Post, en la cual los cálculos reflejan muchos rasgos esenciales de los efectuados en los ordenadores electrónicos reales; en ejemplos elementales se lleva a cabo la enseñanza de los principios de programación en la máquina de Post y se aclaran las posibilidades de ésta, que resultan bastante amplias a pesar de su extrema simplicidad.

Se presupone que el lector no posee casi ningunos conocimientos matemáticos que superen los obtenidos en la escuela primaria.

El autor abriga esperanzas de que el presente libro, en cierta medida, contribuirá al avance de semejantes conceptos como «algoritmo», «computadora universal» y «programación» en la escuela secundaria, incluso en sus grados primarios. La experiencia personal del autor le convence de que los alumnos de los primeros grados, incluso los preescolares mayores, pueden realizar sin dificultad los «cálculos»[1]) en la máquina de Post ateniéndose a un programa prefijado, por ejemplo, mediante una cinta de papel pautada en células y sujetapapeles o botones en calidad de marcas, así como componer simples programas (que no contengan instrucciones de saltos del control). Precisamente por esto en el primer capítulo, más accesible para los escolares menores, está incluido un párrafo especial, el quinto, «Notas metodológicas».

Hace más de cuarenta años Emil L. *Post*, eminente matemático estadounidense, publicó en la «Revista de la lógica sim-

---

[1]) La palabra «cálculos» viene entre comillas, porque no es obligatoriamente en absoluto que los datos iniciales y los resultados de las transformaciones realizadas en la máquina sean números; en una serie de casos son mucho más demostrativas las operaciones con combinaciones de símbolos que no tienen valores numéricos. (Esta nota y las que vienen en adelante pertenecen al autor de la obra.—*N. del T.*)

bólica» («The Journal of Symbolic Logic») el artículo «Procesos combinatorios finitos, enunciación 1» («Finite combinatory processes — formulation 1»), cuya traducción del inglés se adjunta al texto fundamental del presente libro. En este artículo y en el «Acerca de los números computables con la aplicación al problema de decidibilidad» («On computable numbers, with an application to the Entscheidungsproblem») de A. M. *Turing*, matemático inglés, que apareció al mismo tiempo en las «Obras de la sociedad matemática de Londres» («Proceedings of the London Mathematical Society») fueron dadas las primeras precisiones del concepto «algoritmo» que es uno de los centrales en la lógica matemática y cibernética y que empieza a desempeñar un papel de creciente importancia en las cuestiones de automatización y, por eso, en toda la vida de la sociedad contemporánea.

Las precisiones del concepto «algoritmo», propuestas por Post y Turing, no pierden su importancia hasta nuestro tiempo. La máquina de Turing se emplea constantemente como un aparato de servicio en la teoría moderna de algoritmos; la máquina de Post es menos conocida, a pesar de que—o bien precisamente porque—ésta es más sencilla que la primera[1]). Dichas obras son notables también por el hecho de que en ellas unos años antes de la aparición de los primeros ejemplares de las grandes computadoras (llamadas «universales») en funcionamiento (al principio, incluso no electrónicos, sino electromecánicos) fueron anticipados en una forma abstracta los ragos fundamentales de principio de tales máquinas. Las propias estructuras, propuestas por Post y Turing, fueron enunciadas por ellos en forma de ciertas «máquinas abstractas»; esto fue hecho en forma explícita por Turing e implícita, por Post, el cual no emplea el término «máquina». La exposición de las construcciones de Turing se aduce con frecuencia en la literatura dedicada a la teoría de algoritmos, inclusive también en la de divulgación. En lo que se refiere a las construccio-

---

[1]) La máquina de Post tiene estructura más simple que la de Turing, en lo que se refiere a que sus acciones elementales son de mayor simplicidad que las de la máquina de Turing, y los medios de registro son de menor variación, sin embargo, precisamente por estas causas el registro y procesamiento de la información en la máquina de Post exige, hablando en general, mayor volumen de la «memoria» y mayor cantidad de pasos que en la máquina de Turing.

nes de Post, a pesar de ser más simples que las de Turing, ellas, excepto el artículo original de Post, mucho tiempo no se publicaron en la literatura especial, ni en la de divulgación[1]). Al mismo tiempo precisamente estas construcciones, como lo muestra la experiencia de enseñanza del autor, pueden constituir, con no menor razón quo las máquinas de Turing, la introducción natural a la teoría de algoritmos.

Precisamente al concepto del algoritmo, propuesto por Post, está dedicado el presente libro. La exposición está algo modificada y, en particular, se efectúa en forma de descripción de cierta máquina calculadora abstracta. Justamente por eso parece conveniente introducir el propio término de «máquina de Post» y, en general, hacer uso de la terminología que difiere de la de Post

---

La base de este fascículo de las «Lecciones populares de matemáticas» está constituida por las conferencias pronunciadas por el autor para los escolares (en la Universidad Estatal de Moscú M. V. Lomonósov en octubre de 1962, en la escuela-internado adjunta a la Universidad Estatal de Novosibirsk en junio de 1963, en el Palacio de Pioneros de Moscú en diciembre de 1964), así como por las conferencias para los estudiantes de las facultades mecánico-matemática y de filología de la Universidad Estatal de Moscú M. V. Lomonósov a partir del año lectivo de 1961/62.

---

[1]) En el 1967 en la revista «Matemáticas en la escuela» (en ruso), N°N° 1...4, el autor publicó una serie de cuatro artículos que sirvieron como base del presente libro.

CAPÍTULO I. CÓMO TRABAJA LA MÁQUINA DE POST

§ 1. "VISTA EXTERIOR" DE LA MÁQUINA DE POST

Antes de todo advertimos al lector que la máquina de Post no es un dispositivo que existe en realidad y está hecho por alguien; por esta razón las palabras «vista exterior» están puestas entre comillas. La máquina de Post, como también su pariente cercano—la máquina de Turing—representa de por sí una estructura mental que sólo existe en nuestra imaginación[1]) (aunque ella, en principio, podría ser construida «en metal»[2])). Precisamente esto se tiene en cuenta cuando se dice de las máquinas de Post y de Turing, que éstas son máquinas computadoras «abstractas». Sin embargo, para nosotros tendrá poca importancia que la máquina de Post no existe en realidad. Al contrario, supondremos, para mayor evidencia, que «existe de hecho». Y de modo semejante a la posibilidad de aprender el cálculo en el ábaco o en la regla de cálculo, sin tener ante la vista estos instrumentos, tan sólo utilizando sus descripciones e imaginándolos mentalmente, de esta misma manera nosotros aprenderemos a computar en la máquina de Post, concentrando nuestra imaginación en la descripción que ofrecemos a continuación.

La máquina de Post consta de la *cinta* y del *carro* (que se llama también *cabezal de lectura y de registro*). La cinta es infinita y se divide en *células* de igual dimensión: para la evidencia consideraremos que la cinta está dispuesta horizontalmente (fig. 1).

La infinidad de la cinta se encuentra en contradicción con la afirmación hecha anteriormente de que en principio la

[1]) Por esta razón parecen anecdóticas tales, por ejemplo, recomendaciones respecto a la familiarización con las máquinas de este tipo: «para representar mejor el funcionamiento real de la máquina, el interesado debe dirigirse a la literatura técnica» (*Popov A. I.*, Introducción a la lógica matemática, Leningrado, Editorial «Universidad Estatal de Leningrado», 1959, p. 91 (en ruso)).

[2]) Un dispositivo, que permite simular el funcionamiento de la máquina de Post en el caso de pequeños programas y volúmenes del cómputo, fue fabricado en 1970 en la Universidad Estatal de Simferopol (véase *Kasatkin V. N.*, Siete problemas de cibernética, Kiev, 1975, p. 26, (en ruso)).

máquina de Post podría ser construida. El asunto consiste en que hemos declarado que la cinta es **infinita** solamente para sencillez de exposición. Con el mismo éxito podría ser supuesto que la cinta no es infinita, sino que *crece indefinidamente* por ambos lados: por ejemplo, podríamos considerar que la cinta crece en una sola célula, en cuanto el carro llega hasta el

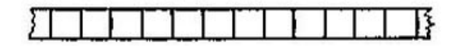


FIG. 1. La cinta de la máquina de Post se divide en células y se extiende infinitamente a la izquierda y a la derecha

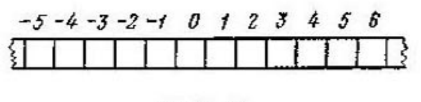


FIG. 2.

fin de la cinta y debe seguir moviéndose (acerca del movimiento del carro véase más abajo), o bien podríamos considerar que por cada unidad de tiempo a la izquierda y a la derecha crece en una célula. Sin embargo, para nosotros será mas cómodo considerar que todas las células a la izquierda y a la derecha ya han crecido y, por eso, aunque en perjuicio de la realidad, suponer que la cinta es infinita por ambos lados.

El orden en que están dispuestas las células es semejante al orden en que se encuentran todos los números enteros. Por consiguiente, es natural introducir en la cinta el «sistema de coordenadas de números enteros», numerando las células mediante los números enteros ..., —3, —2, —1, 0, 1, 2, 3, . . . (fig. 2).

Consideraremos que el sistema de coordenadas está vinculado rígidamente con la cinta y obtendremos de tal modo la posibilidad de indicar una célula cualquiera de la cinta, nombrando su número de orden o coordenada. (A veces, en lo que cabe, es más cómodo introducir a la par con el sistema de coordenadas fundamental, «constante» también el auxiliar, «temporal», desplazado respecto al sistema inicial.)

En cada célula de la cinta puede ya sea no estar escrito nada (tal célula se denomina *vacía*), o bien escrita la *marca* V (en este caso, la célula se llama *marcada*) (fig. 3).

La información acerca de qué células están vacías y cuáles marcadas constituye el *estado de la cinta*. Con otras palabras,

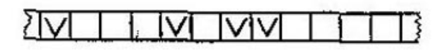


FIG. 3. Cada célula de la cinta está vacía o contiene una marca

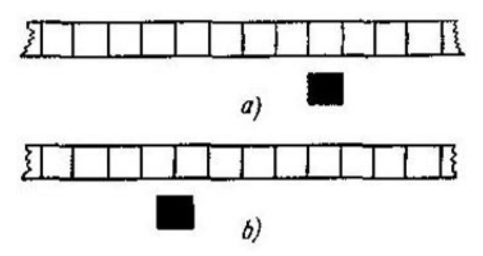


FIG. 4. Cuando el carro está inmóvil, éste se encuentra frente a una de las células de la cinta de la manera que se muestra en la fig. *a*), pero no de la manera que se muestra en la fig. *b*). La situación que viene expuesta en la fig. *b*) puede surgir sólo durante el proceso de movimiento del carro

el estado de la cinta es la distribución de las marcas por sus células [1]). Como veremos a continuación, el estado de la cinta varía en el proceso de funcionamiento de la máquina.

El carro puede desplazarse a lo largo de la cinta a la izquierda y a la derecha. Cuando el carro está inmóvil, éste se encuentra exactamente frente a una sola célula de la cinta (fig. 4, *a*; en este y en los siguientes planos el carro viene representado en forma de un cuadrado ennegrecido); se dice que el carro *observa* esta célula o bien la mantiene en el *campo visual*.

[1]) En el lenguaje matemático exacto el estado de la cinta es una función que a cada uno de los números (número de la célula) pone en concordancia ora la marca, ora, digamos, la palabra «vacío».

La información sobre qué células están vacías y cuáles marcadas y dónde se halla el carro forma el *estado de la máquina de Post*. Por lo tanto, el estado de la máquina se compone del estado de la cinta y de la indicación del número de aquella célula que se observa por el carro. Cada unidad de tiempo (la que llamaremos *paso*) el carro puede desplazarse a una célula a la izquierda o bien a la derecha. Además, el carro puede poner (imprimir) o eliminar (vaciar) la marca en aquella célula frente a la cual se encuentra, así como identificar si hay o no marca en la célula observada por el mismo. En el párrafo 3 será explicado qué es lo que define el funcionamiento del carro, así como qué significa «discernir» (indentificar) en lo que a éste se refiere.

§ 2. PROGRAMA PARA LA MÁQUINA DE POST

El funcionamiento de la máquina de Post consiste en que el carro se desplaza a lo largo de la cinta e imprime o vacía (borra) las marcas. Este trabajo transcurre según instrucciones de determinado aspecto que se denominan programa. Para la máquina de Post se pueden elaborar diversos programas. Vamos a ver la estructuración del programa.

Cada uno de los programas de la máquina de Post consta de instucciones. Llamaremos *instrucción de la máquina de Post* la expresión que tiene uno de los siguientes seis aspectos (las letras i, j, j_1 y j_2 significan por doquier los números naturales 1, 2, 3, 4, 5, . . .):

Primer aspecto. Instrucciones de movimiento a la derecha.

$$i. \Rightarrow j$$

Segundo aspecto. Instrucciones de movimiento a la izquierda.

$$i. \Leftarrow j$$

Tercer aspecto. Instrucciones de impresión de la marca.

$$i. \vee j.$$

Cuarto aspecto. Instrucciones de vaciado (borrado) de la marca.

$$i. \xi\, j$$

Quinto aspecto. Instrucciones de salto del control.

$i.\ ?\!\begin{smallmatrix}\nearrow j_1\\ \searrow j_2\end{smallmatrix}$

Sexto aspecto. Instrucciones de parada.

$$i.\ \text{stop}.$$

Por ejemplo,

$$137.\ \Rightarrow 1$$

es la instrucción de movimiento a la derecha,

$25.\ ?\!\begin{smallmatrix}\nearrow 32\\ \searrow 25\end{smallmatrix}$

es la instrucción de salto del control, mientras que

$$6386.\quad \text{stop}$$

es la instrucción de parada.

El número i, que se encuentra en el comienzo de la instrucción, se denomina *número* de instrucción. Así, en las instrucciones que acabamos de aducir sus números son 137, 25 y 6386, respectivamente. El número j, que culmina la instrucción (asimismo en las instrucciones de salto del control, cada uno de los números j_1 y j_2), lo llamaremos *salto* (con ello, en la instrucción de salto del control j_1 es el salto superior y j_2, el salto inferior). Las instrucciones de parada no tienen salto. Así, en las instrucciones que acabamos de citar en calidad de saltos sirven los números 1, 32 y 25, con la particularidad de que el número 32 es el salto superior y 25, el inferior.

Denominarémos *programa para la máquina de Post* la lista finita no vacía (es decir, la que contiene aunque sólo sea una instrucción) de las instrucciones de la máquina de Post, que posee las dos propiedades siguientes:

1) En primer lugar en esta lista se encuentra la instrucción con el número 1, en el segundo (si existe), la instrucción con el número 2, etc.; en general, en el k-ésimo lugar está la instrucción con el número k.

2) El salto de cualquiera de las instrucciones que figuran en la lista coincide con el número de cierta instrucción (que puede ser otra o la misma) en aquélla (con mayor precisión:

para cada uno de los saltos de cada una de las instrucciones de la lista se encontrará en ésta tal instrucción, cuyo número es igual al salto examinado).

Por ejemplo, la siguiente lista será el programa para la máquina de Post:

1. stop　　3. ξ 3

2. ? $\begin{smallmatrix} \nearrow 4 \\ \searrow 1 \end{smallmatrix}$　　4. stop

En tanto que las siguientes dos listas no servirán de programas para la máquina de Post, aunque están compuestas de las instrucciones para la misma:

2. ? $\begin{smallmatrix} \nearrow 4 \\ \searrow 1 \end{smallmatrix}$　　3. ξ3

1. stop　　4. stop

(no está cumplida la primera condición);

1. stop　　3. ξ 3

2. ? $\begin{smallmatrix} \nearrow 4 \\ \searrow 5 \end{smallmatrix}$　　4. stop

(no está cumplida la segunda condición).

Para mayor evidencia escribiremos en columna los programas para la máquina de Post. El número de instrucciones del programa se llama longitud del programa.

Ejercicio. Copiar todos los programas para la máquina de Post de la longitud 1. ¿Cuántos programas existen de la longitud 2, de la longitud 3, de la longitud n?

§ 3. FUNCIONAMIENTO DE LA MÁQUINA DE POST

Para que la máquina de Post comience a trabajar es necesario prefijar, primeramente, cierto programa y, en segundo lugar, cierto estado de ella, es decir, distribuir de cualquier modo las marcas por las células de la cinta (en particular, se pueden dejar todas las células vacías) y colocar el carro frente a una

de las células. Como regla, supondremos que en el estado inicial (o sea, en el prefijado primeramente) de la máquina el carro siempre se instala frente a la célula con el número (coordenada) cero. Llegando a este acuerdo, el estado inicial de la máquina queda definido completamente por el estado de la cinta.

Como ya se dijo, el programa es tal instrucción basándose en la cual trabaja la máquina. El funcionamiento de la máquina a base del programa prefijado (también con el estado inicial asignado) transcurre del modo siguiente. La máquina se pone en el estado inicial y comienza a cumplir la primera instrucción del programa (qué significa «cumplir la instrucción» se explicará más adelante). Esta instrucción se cumple dando un paso, después de lo que la máquina empieza a cumplir aquella instrucción, cuyo número (lo llamaremos α) es igual al salto (a uno de los saltos, si éstos son dos) de la primera instrucción. Ésta también se cumple en un paso, después de lo que comienza a cumplirse la instrucción, cuyo número es igual al salto de la instrucción con el número α. En general, cada instrucción se cumple en un solo paso, mientras que el tránsito después de haber ejecutado una instrucción, al cumplimiento de la otra, transcurre según la siguiente regla: sea que en el k-ésimo paso se cumplía la instrucción con el número i, en este caso, si ésta tiene el único salto j, entonces durante el $k+1$-ésimo paso se cumple la instrucción con el número j; si ésta tiene dos saltos j_1 y j_2, entonces en el $k+1$-ésimo paso se cumple una de las dos instrucciones, es decir, con el número j_1 o bien con el número j_2 (cuál, precisamente, se indicará más adelante); si, por último, la instrucción que se ejecuta en el k-ésimo paso no tiene salto absolutamente, entonces en el $k+1$-ésimo paso y durante todos los pasos posteriores ninguna instrucción se cumple: la máquina se para. Queda por explicar, qué significa cumplir la instrucción y cuál de los saltos, cuando hay dos, se elige como número de la instrucción siguiente.

La ejecución de la instrucción de movimiento a la derecha consiste en que el carro se desplaza a una célula en dicha dirección. El cumplimiento de la instrucción de movimiento a la izquierda consiste en que el carro se desplaza a una célula en la dirección indicada. El procedimiento para ejecutar la instrucción de impresión de la marca consiste en que el carro

la señala en la célula observada; esta instrucción se puede cumplir sólo en caso de que la mencionada célula, antes de ejecutar la instrucción, esté vacía; pero si en la misma ya se encuentra la marca, la instrucción se considera irrealizable. El cumplimiento de la instrucción para borrar la marca consiste en que el carro la elimina en la célula observada; esta instrucción puede ejecutarse sólo en caso de que dicha célula esté marcada; sin embargo, si la célula en cuestión no contiene marca, la instrucción se considera irrealizable. El cumplimiento de la instrucción de salto del control con el salto superior j_1 y el inferior j_2 no cambia de manera alguna el estado de la máquina: ninguna de las marcas se elimina y se imprime, además el carro queda inmóvil (la máquina realiza, por así decirlo, un «paso sin movimiento»); no obstante, si la célula observada estaba vacía, antes de comenzar el cumplimiento de esta instrucción, entonces acto seguido debe ejecutarse la instrucción con el número j_1, si por el contrario, esta célula contenía la marca, la siguiente instrucción que deberá realizarse será la del número j_2 (el papel de la instrucción de salto del control se reduce, por consiguiente, a que el carro al ejecutarla parece como si «identificara» que la marca se encuentra ante él, precisamente esto es lo que se tenía en cuenta en la penúltima frase perteneciente al párrafo 1). El cumplimiento de la instrucción de parada tampoco cambia de ninguna manera el estado de la máquina y consiste en que la máquina se para.

Si ahora, después de prefijar el programa y cualquier estado inicial, se pone en marcha la máquina, se realizará una de las tres variantes siguientes:

1) Durante la ejecución del programa la máquina llegará al cumplimiento de una instrucción irrealizable (impresión de la marca en una célula no vacía o borrado de la marca en la célula vacía); con ello, cesa el cumplimiento del programa, la máquina se para; transcurre la llamada *parada sin resultados.*

2) Al realizar el programa, la máquina alcanzará el cumplimiento de la instrucción de parada; en este caso, el programa se considera ejecutado, la máquina se para; transcurre la llamada *parada de resultados.*

3) Durante la ejecución del programa la máquina no llegará hasta el cumplimiento de ninguna de las instrucciones indicadas en las dos primeras variantes; con ello, el cumplimiento

del programa nunca cesa, la máquina nunca se para; el proceso de funcionamiento de la máquina transcurre infinitamente.

§ 4. EJEMPLOS DEL CUMPLIMIENTO DE LOS PROGRAMAS

Vamos a prefijar, por ejemplo, el estado inicial indicado en la fig. 5 y el siguiente programa:

1. $\vee 4$ 4. $\Rightarrow 5$
2. $\xi\ 3$
3. $\Leftarrow 2$ 5. $? \begin{smallmatrix} 4 \\ 3 \end{smallmatrix}$

Veamos cómo trabajará la máquina con tal estado inicial y tal programa.

En el primer paso se ejecutará la instrucción N°1. Después del primer paso el estado de la máquina será el que se indica

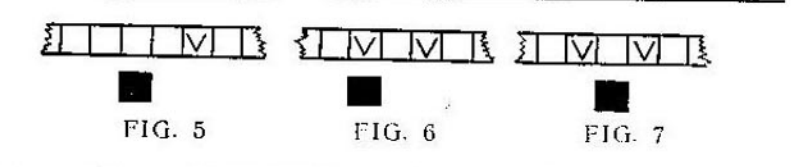


FIG. 5 FIG. 6 FIG. 7

en la figura 6. Una vez realizada la instrucción N°1 es necesario pasar al cumplimiento de aquella instrucción, cuyo número coincide con el salto de la instrucción N°1, es decir, a realizar la instrucción N°4. Ésta se ejecutará durante el segundo paso y el estado de la máquina será el de la figura 7. Ahora hay que cumplir la instrucción N°5 (ya que el salto de la instrucción N°4 es igual a 5). Ésta se realizará en el tercer paso, a resultas del cual el estado de la máquina no cambiará y quedará tal como se muestra en la figura 7. En vista de que la célula observada en este caso está vacía, a continuación, debe cumplirse la instrucción, cuyo número es igual al salto superior, es decir, al número 4. Una vez ejecutada durante el cuarto paso la instrucción N°4, la máquina llegará al estado que se indica en la figura 8. Ahora, en el quinto paso, se cumplirá la instruc-

ción N°5. Esta vez la célula observada está marcada, por este motivo, acto seguido, será ejecutada la instrucción con el número que es igual al salto inferior, es decir, al número 3. Después de cumplir en el sexto paso la instrucción N°3 la máquina llegará al estado que se muestra en la figura 9 y comenzará, en el séptimo paso, el cumplimiento de la instrucción N°2. Sin embargo, esta instrucción resulta irrealizable,

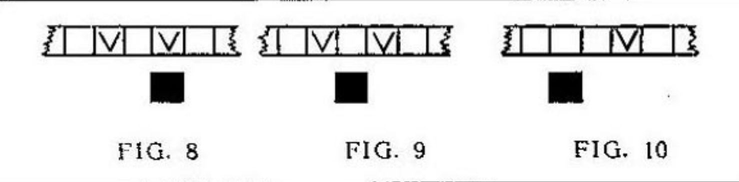


FIG. 8 FIG. 9 FIG. 10

puesto que ordena borrar la marca en la célula vacía. Por consiguiente, durante el séptimo paso ocurrirá una parada sin resultados.

Los diversos programas aplicados al mismo estado inicial pueden llevar a distintos finales: a paradas sin resultados o de resultados, o bien al funcionamiento de la máquina sin paradas (infinito). En efecto, prefijemos, por ejemplo, el estado inicial indicado en la figura 10. Apliquemos a este estado inicial el siguiente programa:

1. ⇒ 2
2. ⇒ 3
3. V.

La máquina realizará dos pasos y en el tercer paso se producirá una parada sin resultados. A este estado inicial vamos a aplicar el programa:

1. ⇒ 2
2. ⇒ 3
3. stop.

La máquina hará dos pasos y, luego, durante el tercero ocurrirá una parada de resultados. Por fin, apliquemos a este mismo estado inicial el programa:

1. ⇒ 1.

La máquina va a trabajar infinitamente. Ahora apliquemos el programa:

1. $? \begin{matrix} \nearrow 1 \\ \searrow 1 \end{matrix}$.

La máquina de nuevo funcionará infinitamente (a pesar de que ni el registro en la cinta, ni la posición del carro, en este caso, se modificarán).

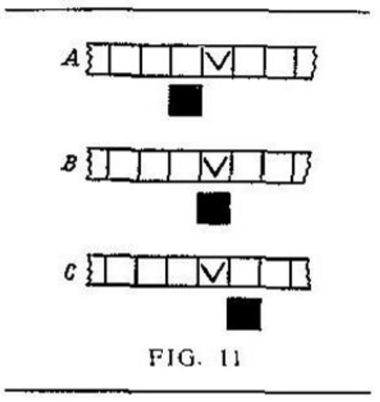


FIG. 11

Precisamente así, el mismo programa, aplicado a diversos estados iniciales, puede producir distintas variantes. Por ejemplo, examinemos el siguiente programa:

1. $? \begin{matrix} \nearrow 4 \\ \searrow 1 \end{matrix}$

2. ξ 3

3. stop

4. $\Rightarrow$ 2.

Empleémoslo para los estados iniciales A, B y C que se ofrecen en la figura 11. Para el estado inicial A obtendremos una parada de resultados en el cuarto paso, para B — el funcionamiento infinito de la máquina y para C — una parada sin resultados en el tercer paso. El uso a los mismos estados iniciales

del programa

1. ? ⟨ 4 / 3 3. stop

4. ⇒ 2

2. V4

produce una parada sin resultados para A, de resultados para B y funcionamiento infinito de la máquina para C.

Ejercicio 1. ¿Puede existir o no un programa que produzca una parada de resultados con cualquier estado inicial? ¿Una parada sin resultados? ¿El funcionamiento ilimitado de la máquina? ¿Cuál será el número mínimo de instrucciones en estos programas?

Ejercicio 2. Acerca de cierto programa es conocido que existe un estado inicial para el cual aquél proporciona una parada de resultados, existe un estado inicial para el cual el primero produce una parada sin resultados y existe un estado inicial para el cual el mismo da el funcionamiento infinito de la máquina. Demostrar que el número de instrucciones en este programa no es menos de **4**. Escribir todos estos programas de la longitud 4.

§ 5. NOTAS METODOLÓGICAS

Este párrafo se dirige a los maestros de la escuela primaria, a los dirigentes de los círculos escolares para los alumnos menores y, en general, a todos los que decidan enseñar la máquina de Post a los escolares de las clases primarias.

Como ya se ha mencionado en el prefacio, se puede enseñar a trabajar en la máquina de Post (representada mediante un dibujo de tiza en la pizarra o bien con ayuda de una cinta de papel y botones o sujetapapeles en calidad de marcas) incluso a los escolares del primer grado. Indudablemente, en la exposición para los escolares menores es preciso no sólo omitir ciertos detalles, sino que introducir nuevos acuerdos. Así, es conveniente

1) no mencionar el sistema de coordenadas en la cinta (éste es necesario sólo para precisar qué es el «estado») y no

introducir el concepto «estado» (para la cinta o para la máquina en conjunto);

2) se puede utilizar en calidad de números de los programas no cifras (o bien no sólo cifras), sino que cualesquier imágenes (por ejemplo, imágenes de las figuras geométricas);

3) suponer que la cinta no es infinita, sino finita, y ponerse de acuerdo que la máquina se para, cuando el carro llega hasta el fin de la cinta;

4) no decir absolutamente nada sobre una parada sin resultados; no obstante, si ésta aparece durante el cumplimiento de cualquier programa, entonces decir que, en este caso, la «máquina se rompe»;

5) introducir no todas las instrucciones de una vez, sino paulatinamente, acompañado la introducción de cada nueva instrucción con ejercicios demostrativos.

Es lógico, que los ejercicios deben ser seleccionados especialmente. Por ejemplo, es útil proponer lo siguiente:

Ejercicio. La máquina comienza desde el estado inicial «vacío» (es decir, tal, con el que todas las células están vacías) y trabaja según el programa:

1. $\vee$ 2
2. $\Rightarrow$ 3
3. $\Rightarrow$ 1;

se pregunta qué ocurrirá con la cinta. La respuesta es bastante demostrativa; ésta se ofrece en la figura 12.

Por lo demás, consideramos que el alumno debe (y puede) saber aplicar cualquier programa a todo estado inicial de



FIG. 12

manera que cualesquier ejercicios que no exigen mucho tiempo para cumplir los programas (incluso para la composición de ciertos programas, por ejemplo, el que conduce al estado de la cinta mostrado en la figura 12) son plenamente convenientes.

Sin duda alguna, en la exposición destinada a los escolares menores, no es conveniente hacer uso de las palabras «algoritmo», «máquina de Post» (así como, en general, la palabra «máquina»; en lugar de las palabras el «carro se traslada a una célula a la derecha» es mejor decir, por ejemplo, «nosotros pasamos a la célula vecina que se encuentra a la derecha», etc.). Es más, la explicación de para qué se necesita la estructura expuesta es útil dejarla hasta el tiempo en que se alcance su entendimiento claro y suficiente libertad en la ejecución de los programas. Aquí es muy oportuno aducir la siguiente cita del epígrafo al capítulo nueve del libro de *W. W. Sawyer* «Prelude to mathematics», Penguin Books, Bristol, 1955, p. 125 («Preludio a las matemáticas», traducción del inglés al ruso, Moscú, Editorial «Prosveschenie», 1965): «Primeramente yo les diré lo que hago y, después, explicaré para qué».

En general, la capacidad de percibir algún sistema de conceptos o bien alguna construcción antes (e independientemente) de recibir una información acerca de para qué esto es necesario, es decir, antes (e independientemente) de cualquiera aplicación, representa para nosotros una de las más importantes cualidades que se educan mediante los estudios de las matemáticas. La idea acerca del objetivo que persigue la exposición de este u otro material es posible que contribuya a su memorización, pero no debe influir sobre el entendimiento de lo que puede y debe tener lugar independientemente de este objetivo. La habilidad de pensar de modo formal es una capacidad especial que se desarrolla, como toda destreza, debida al entrenamiento. Semejante entrenamiento podría empezarse desde edades tempranas. Como elementos accesibles del entrenamiento mencionado para los escolares del primer grado pueden servir también la adición de los números polidígitos (con lo que los números polidígitos se entienden sin la habitual semántica cuantitativa, simplemente como cadenas de cifras, mientras que la suma se determina mediante un algoritmo de adición en columna [1])y sencillos ejercicios con la máquina de Post.

---

[1]) Véase *Uspenski V. A.* Para la enseñanza de las matemáticas en la escuela primaria. Revista «Matemáticas en la escuela» (en ruso), 1966, N° 2, p. 38; se tiene en cuenta que el procedimiento de adición formal, escrito sólo a continuación, se utiliza para obtener la suma de dos cantidades de manzanas o cuadernos.

La conclusión de la exposición de cierta estructura, es decir, la exposición que infaliblemente va acompañada de suficiente cantidad de ejemplos y ejercicios, significa el fin de una etapa importante, después de la cual, durante la siguiente etapa, ya se puede pasar a la aplicación de la estructura descrita. Para la máquina de Post en calidad de esta aplicación sirven los cálculos que pueden efectuarse con su ayuda. Para subrayar el límite que divide estas dos etapas nos restringiremos en este capítulo tan sólo a la primera familiarización con la máquina de Post.

Después de tal conocimiento es oportuno comunicar a los alumnos que la máquina de Post puede utilizarse para obtener resultados de operaciones aritméticas y proponer ejemplos de programas (a continuación también de problemas para elaborar programas) que conduzcan de los números registrados en la cinta de la máquina a los resultados de unas u otras operaciones con los mismos. Entre estas últimas la más simple es la adición de la unidad a un número. Es lógico, que para esto es preciso con anticipación ponerse de acuerdo cómo registrar los números en la cinta de la máquina de Post, así como hacer además una serie de precisiones. Pero, esto es el objeto del siguiente capítulo que se dedicará especialmente a la adición de la unidad. En calidad de un ejemplo útil se puede proponer al lector que él mismo intente a atribuir un sentido exacto a la siguiente enunciación: «Elaborar para la máquina de Post un programa que efectúe la adición de la unidad». ¿Quizá semejantes sentidos resultará haber no sólo uno, sino muchos?

CAPÍTULO II. ADICIÓN DE LA UNIDAD EN LA MÁQUINA DE POST

En la máquina de Post pueden efectuarse diversos cálculos. En el presente capítulo a la atención del lector se propone el más simple de tales cómputos, es decir, la adición de la unidad a un número arbitrario; este cálculo se realiza con distintas variantes que dependen de una u otra enunciación de las condiciones iniciales.

El examen de cómo en la máquina de Post se efectúa la adición de la unidad, a pesar del carácter elemental de este tema, permite familiarizar al lector (es lógico, que en una forma muy simplificada) con problemas que surgen también al operar los ordenadores reales de acción rápida. El hecho consiste en que el problema matemático fundamental que se plantea ante el hombre durante su trabajo en las computadoras resulta ser el mismo tanto para las máquinas reales, como para las «abstractas». Este problema consiste en **elaborar para la máquina el programa que conduzca al objetivo prefijado**: la elaboración de programas semejantes se denomina *programación*. En el presente capítulo se examina el problema (con más exatitud, una serie de problemas) sobre cómo elaborar para la máquina de Post programas que lleven al aumento del número inicial en una unidad.

Además de la comunidad del problema fundamental, muchos otros problemas de la programación para la máquina de Post son característicos también en la programación para las máquinas reales. He aquí algunos de momentos semejantes que se hacen notables ya al examinar el problema sobre la adición de la unidad:

1. Se pueden componer diversos programas que llevarán al objetivo prefijado (a un mismo), o sea, que realizarán el procesamiento prefijado de la información; el lector verá (§ 2, ejercicio) que para la máquina de Post es incluso posible una infinita cantidad de programas que realicen la adición de la unidad (ya para la más simple variante de tal adición).

2. Si se amplifica la clase de los datos iniciales, respecto de los cuales el programa debe llevar al resultado necesario, el

problema acerca de la construcción de semejante programa es más difícil y el propio programa, que sirve como solución del problema, más complicado; el lector verá cómo se complicará el programa de adición de la unidad, al amplificarse la clase de condiciones iniciales tolerables.

3. Al construir los programas que dan la solución de los problemas más generales o más complicados, con frecuencia resulta más cómodo hacer uso, en calidad de bloques estructurales acabados, los programas elaborados antes para problemas más particulares o más simples; el lector verá en el § 4 cómo, al construir el programa para el problema general acerca de la adición de la unidad, pueden utilizarse los programas que dan la solución para problemas más particulares.

4. Al componer el programa se tiene que tomar en consideración no sólo a qué números es necesario aplicarlo, sino que también cómo estos números se disponen en el «dispositivo de memoria», o bien en la «memoria», de la máquina; el lector verá que las variantes del problema acerca de la adición de la unidad se distinguirán una de la otra sólo por la disposición del número inicial respecto al carro.

5. La tendencia de que los programas obtenidos sean, en la medida de lo posible, cortos, es, de ordinario, muy natural, en tanto que para las máquinas y los problemas reales la brevedad de los programas en una serie de casos puede tener el valor decisivo; el lector verá que a la minimización de la longitud del programa se prestará atención especial.

§1 REGISTRO DE LOS NÚMEROS EN LA MÁQUINA DE POST Y PLANTEAMIENTO DEL PROBLEMA ACERCA DE LA ADICIÓN DE LA UNIDAD

En la máquina de Post pueden ser realizadas distintas operaciones con los números. Para ello, es preciso sobre todo ponerse de acuerdo de cómo en la máquina de Post se registrarán los números. Se tratará siempre de números no negativos enteros 0, 1, 2, 3, 4, . . .

Examinemos la sucesión finita, de las células marcadas en la cinta que siguen una tras otra, situada entre dos células vacías. Tal sucesión de las células marcadas la llamaremos *juego*, mientras que el número de células en ésta, *longitud*

del juego. Así, en la figura 13 se muestra un juego de longitud 3, en tanto que en la figura 14, tres juegos: de longitud 5, longitud 1 y longitud 2.

Ahora, convengamos registrar el número n en la cinta mediante el juego de longitud $n + 1$ y llamar el propio juego

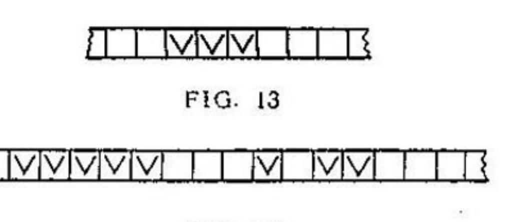


FIG. 13

FIG. 14

registro de máquina del número n. Por consiguiente, en las figuras 13 y 14 se ofrecen los registros de máquina de los números 2, 4, 0 y 1.

Prefijemos el objetivo de realizar en la máquina de Post la adición de la unidad. Entenderemos nuestro problema del modo siguiente: es necesario elaborar un programa que al ser aplicado a la cinta, en la cual está registrado el número arbitrario n, conduzca a la parada de resultados, con la particularidad de que después de la parada en la cinta debe estar registrado el número $n + 1$ (se tiene en cuenta la composición de **un solo** programa que sirva para **cualquier** n).

En la frase anterior el problema está planteado aún no con plena precisión, ya que ni sobre el estado inicial (es decir, prefijado en el comienzo), ni sobre el final (es decir, que surge después de la parada de resultados) de la máquina no se dice dónde precisamente está registrado el número, ni qué más hay registrado en la cinta; tampoco se dice dónde debe encontrarse el carro. Supongamos que al principio y al final del funcionamiento del programa en la cinta vienen registrados sólo los números correspondientes (n al comenzar y $n + 1$ al finalizar), dispuestos en un lugar arbitrario de la cinta, mientras que la cinta restante está vacía. Con objeto de facilitar nuestro problema no aplicaremos ningunas limitaciones adicionales en el estado final: por lo tanto, nos convendrá cualquier pro-

grama que proporcione la cinta con el registro del número $n + 1$, dondequiera que este registro se encuentre y en cualquier parte que esté el carro. Al mismo tiempo, haremos suposiciones cada vez más amplias con respecto a la disposición recíproca del carro y del registro de máquina del número en el estado inicial de ésta. Por consiguiente, tendremos que tratar no un problema único, sino toda una serie de ellos. Con insistencia aconsejamos al lector, antes de leer la solución de cualquier problema, tratar de resolverlo por sí mismo.

§ 2. ADICIÓN DE LA UNIDAD EN EL CASO MÁS SIMPLE

Empecemos por la más fuerte limitación, aplicada a la disposición recíproca del registro de máquina del número y del carro al comienzo, y, del mismo modo, por el problema más simple. Lo denominaremos problema 1.

Problema 1 (enunciación larga). Se requiere escribir el programa para la máquina de Post que posea la siguiente propiedad. Cualquiera que sea el número n, si el estado inicial de la máquina de Post es tal, que en la cinta está registrado el número n (mientras que la cinta restante está vacía) y el carro se encuentre frente a la célula más a la izquierda del registro, el cumplimiento del programa debe llevar a la parada de resultados, después de la cual en la cinta (en un lugar arbitrario) debe estar inscrito el número $n + 1$ (en tanto que la cinta restante debe estar vacía), con la particularidad de que el carro puede encontrarse dondequiera.

Designemos por A_n el conjunto de todos aquellos estados de la máquina de Post, en cada uno de los cuales en la cinta vienen marcadas exactamente $n + 1$ células, mientras que el carro se encuentra frente a la célula más izquierda de entre las marcadas. En la fig. 15 se ofrecen varios estados de la clase A_2; éstos difieren uno del otro sólo en que el juego y el carro, «ligado con éste rígidamente», ocupan distintas posiciones con respecto al origen de coordenadas.

Designemos por E_n el conjunto de todos aquellos estados de la máquina de Post, en cada uno de los cuales vienen marcadas en la cinta exactamente $n + 1$ células y el carro puede encontrarse dondequiera. Al conjunto E_n pertenecen todos los esta-

dos de A_n y aún muchos otros. En la fig. 16 se muestran varios estados de la clase E_2.

Ahora, el problema 1 puede ser enunciado con más brevedad:

Problema 1 (enunciación corta). Escribir tal programa para la máquina de Post que para cualquier n siendo aplicado al estado arbitrario de la clase A_n, da la parada de resultados en cierto estado de la clase E_{n+1}.

Antes de proceder a resolver el problema 1, notemos que en su planteamiento no se dice nada acerca de qué debe obtenerse

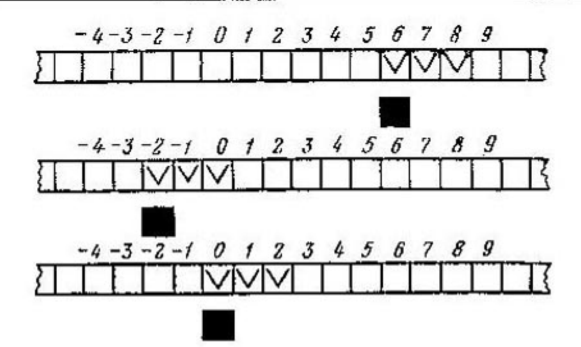


FIG. 15. Los estados de la clase A_n se distinguen uno del otro sólo por el «desplazamiento» con respecto al sistema de coordenadas

al aplicar el programa buscado a los estados que no pertenecen a ninguna de las clases A_n ($n = 0, 1, 2, \ldots$); esto significa que para nosotros no importa qué sucederá con semejantes estados elegidos en calidad de iniciales: nos convendrá cualquier programa que traslada los estados de A_n a los de E_{n+1}, hiciera lo que hiciese éste con los estados restantes.

Será solución del problema 1, por ejemplo, tal programa:

Programa I_1

1. $\Leftarrow$ 2
2. $\vee$ 3
3. stop.

Hemos escrito «por ejemplo», porque la solución del problema 1 no es la única: también son posibles otros programas

que satisfagan el planteamiento del problema. Por ejemplo, el siguiente programa también será la solución del problema 1:

1. ⇒2   3. ⇐4

4. ⇐5

2. ? ⟨ 3, 3   5. V 6

6. stop.

Sin embargo, el programa I_1 escrito anteriormente será el más corto (más exactamente, uno de los más cortos) de los progra-

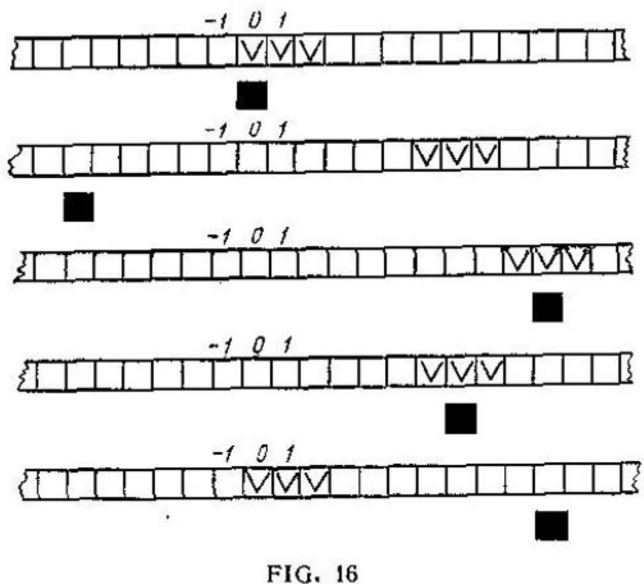


FIG. 16

mas que satisfacen el planteamiento del problema 1. En efecto, se puede demostrar (y nosotros aconsejamos al lector que lo haga) que ningún programa de la longitud 1 ó 2 puede servir como solución del problema 1; sin embargo, existe aún exactamente un programa de la longitud 3 que también es la solución de nuestro problema; he aquí este programa:

Programa I_2

1. $\Leftarrow$ 3
2. stop
3. $\vee$ 2.

Ejercicio. Demuestren que existen infinita cantidad de programas que sirven de soluciones para el problema 1.

Observación. Con n fijado, los estados de A_n se distinguen uno del otro sólo por el desplazamiento con respecto al origen de coordenadas (fig. 15). Cualquiera que sea el programa que hubiéramos aplicado a estos estados iniciales, los resultados que se obtendrían, evidentemente, se distinguirían uno del otro por un mismo desplazamiento. Por esta razón, es suficiente seleccionar de cada clase A_n un solo estado a_n y componer el programa que va a trasladar cada a_n a cierto estado de E_{n+1}. Cualquier semejante programa automáticamente será la solución del problema 1. Notemos, que el estado a_n se puede seleccionar de A_n de modo en absoluto al azar.

De manera completamente análoga al problema 1 puede ser resuelto el problema 1′ que difiere del primero sólo en que al comienzo el carro observa la célula más a la derecha del juego. Precisamente, designemos por A'_n la clase de todos aquellos estados de E_n, en los cuales el carro observa la célula más derecha de entre las marcadas. En este caso la enunciación corta del problema 1′ será como sigue:

Problema 1′ (enunciación corta). Escribir tal programa para la máquina de Post que para cualquier n, siendo aplicado al estado arbitrario de la clase A'_n, da la parada de resultados en cierto estado de la clase E_{n+1}.

Los siguientes dos programas serán los más cortos y únicos que satisfacen el planteamiento del problema 1′.

Programa I'_1

1. $\Rightarrow$ 2
2. $\vee$ 3
3. stop.

Programa I'_2

1. $\Rightarrow$ 3
2. stop
3. $\vee$ 2.

§ 3. ADICIÓN DE LA UNIDAD EN CASOS MÁS COMPLICADOS

Ahora no exigiremos que al comienzo el carro se encuentre obligatoriamente frente a una de las células extremas del juego, como en los problemas 1 y 1'. Limitémonos a la exigenica de que en el estado inicial el carro observe una de las células del juego.

Problema 2 (enunciación larga). Se requiere escribir el programa para la máquina de Post que posea la siguiente

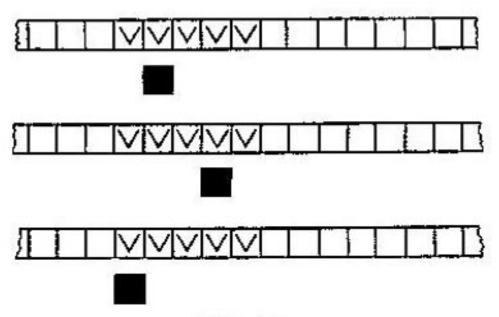


FIG. 17

propiedad. Cualquiera que sea el numero n, si el estado inicial de la máquina de Post es tal, que en la cinta hay registro de máquina del número n (mientras que la cinta restante está vacía) y el carro se encuentra frente a una de las células del registro, entonces la ejecución del programa debe llevar a la parada de resultados, después de lo cual en la cinta (en un sitio arbitrario) debe ser registrado el número $n+1$ (en tanto que la cinta restante debe estar vacía), con la particularidad de que el carro puede encontrarse dondequiera.

Designemos por B_n el conjunto de todos aquellos estados de la máquina de Post, en cada uno de los cuales vienen marcadas en la cinta exactamente $n+1$ células, mientras que el carro está frente a una de las células marcadas. Es evidente, que la clase B_n es una parte de la clase E_n y la primera contiene en calidad de parte integrante la clase A_n. En la fig. 17 se exponen varios estados de la clase B_4 y en la fig. 18, la vista gene-

ral del estado de la clase B_n. En este caso, obtendremos la siguiente enunciación corta del problema 2:

Problema 2 (enunciación corta). Escribir tal programa para la máquina de Post que para cualquier n, siendo aplicado al estado arbitrario de la clase B_n, da la parada de resultados en cierto estado de la clase E_{n+1}.

Es evidente, que cada solución del problema 2 será simultáneamente también la solución del problema 1 y del problema 1'. Al mismo tiempo existen soluciones del problema 1 que no sirven para el problema 2. Tales son, por ejemplo, los

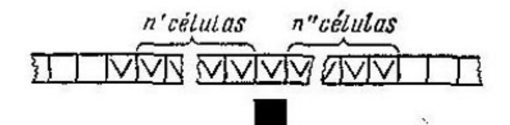


FIG. 18. Vista general del estado de la clase B_n. Aquí $n' \geqslant 0$, $n'' > 0$, $n' + n'' = n$

programas I_1 y I_2. Para el problema 2, como también para el 1, existe un conjunto infinito de programas que son su solución. Como antes, nos interesarán los programas más cortos. Se puede demostrar (recomendamos al lector que lo haga) que ningún programa de la longitud 1, 2 ó 3 puede ser la solución del problema 2. Al mismo tiempo se pueden obtener soluciones del problema 2 de la longitud 4. He ahí una de ellas:

Programa II_1

1. $\Leftarrow 2$
2. $? \begin{matrix} \nearrow 3 \\ \searrow 1 \end{matrix}$
3. $\vee\ 4$
4. stop.

Ejercicio 1. Hallen dos soluciones más del problema 2, que tengan la longitud 4 y contengan una instrucción de movimiento a la izquierda. Verifiquen que cada solución de las halladas por Uds. contiene la instrucción de impresión de la marca, la de salto del control y la de parada.

Ejercicio 2. Demuestren que además del programa Π_1 existen aún exactamente once (Π_2, Π_3, . . ., Π_{12}) soluciones del problema 2 que tienen la longitud 4 y contienen la instrucción de movimiento a la izquierda. Apunten estas soluciones.

Ejercicio 3. Comprueben que los programas Π'_1, Π'_2, ..., Π'_{12} que se obtienen de los programas Π_1, Π_2, . . ., Π_{12} mediante el cambio del signo $\Leftarrow$ por el $\Rightarrow$ también son soluciones del problema 2. Demuestren que los programas Π_1, Π_2, . . ., Π_{12}, Π'_1, Π'_2, . . ., Π'_{12} agotan todas las soluciones del problema 2 de la longitud 4.

Ahora, examinemos aquellos estados iniciales en los cuales el carro mantiene en el campo visual no cualquiera de las células del juego inicial, sino que cierta célula vacía. Con ello,

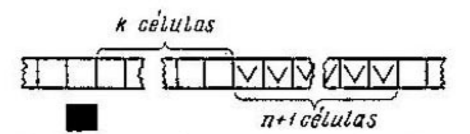


FIG. 19. Vista general del estado de la clase C_n. Aquí $k \geqslant 0$

supondremos, que es sabido con anterioridad, que el carro se encuentra a la izquierda o a la derecha del juego inicial. Las palabras «es sabido con anterioridad» significan que se exige hallar no un programa único, que trabaja en todos los casos, sino que dos programas, uno de los cuales trabaja en el caso, si el carro inicialmente se encuentra a la izquierda del juego, mientras que el otro, en el caso, cuando el carro al principio está a la derecha del juego. Por lo tanto, tenemos aquí no sólo uno, sino que dos problemas. Para obtener en seguida las enunciaciones cortas de estos problemas, introduzcamos las siguientes designaciones. Designemos por C_n el conjunto de todos aquellos estados de la clase E_n, en los cuales el carro se encuentra a la izquierda del juego, y por C'_n, el conjunto de todos los aquellos estados de la clase E_n, en los cuales el carro está a la derecha del juego. En la fig. 19 se muestra la vista general del estado de la clase C_n, en tanto que en la fig. 20, la vista general del estado de la clase C'_n.

Problema 3 (enunciación corta). Escribir tal programa para la máquina de Post que para cualquier n, siendo aplicado al

estado arbitrario de la clase C_n, da la parada de resultados en cierto estado de la clase E_{n+1}.

Problema 3′ (enunciación corta). Escribir tal programa para la máquina de Post que para cualquier n, siendo aplicado al estado arbitrario de la clase C'_n, da la parada de resultados en cierto estado de la clase E_{n+1}.

Es evidente, que cada programa que es la solución del problema 3 puede ser convertido en la solución del problema 3′,

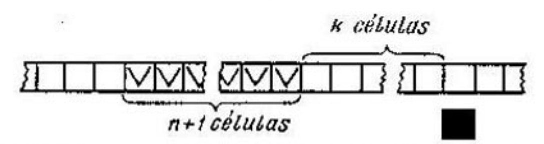


FIG. 20. Vista general del estado de la clase C'_n. Aquí $k \geqslant 0$

si cambiamos en todas partes el signo ⇒ por el ⇐, mientras que el signo ⇐, por el ⇒. Mediante la misma operación cualquiera solución del prblema 3′ se transforma en la solución del problema 3. Por lo tanto, es suficiente resolver sólo uno de estos problemas. He aquí una de las soluciones del problema 3:

Programa III

1. ⇒ 2
2. ? ↗1 ↘3
3. ⇐ 4
4. V 5
5. stop.

Intenten demostrar que el problema 3 no tiene soluciones más cortas. ¿Cuántas soluciones de este problema existen de la longitud 5?

§ 4. ADICIÓN DE LA UNIDAD EN UN CASO AÚN MÁS COMPLICADO

En este párrafo examinaremos en calidad de clase de los estados iniciales (de partida) la unión de todas las clases B_0, C_0, B_1, C_1, B_2, C_2, B_3, C_3, . . . del párrafo anterior. Por consi-

guiente, esta clase será compuesta de todos aquellos, y sólo de aquellos, estados de la máquina, en cada uno de los cuales el carro se encuentra ora frente a cierta célula del juego inicial, ora a la izquierda del juego.

Designemos por D_n el conjunto de todos aquellos estados de la clase E_n, en los cuales la célula observada por el carro bien esté marcada, o bien esté dispuesta a la izquierda de todas las células marcadas. Por lo visto, para cada n la clase D_n es la unión de las clases B_n y C_n.

Problema 4. Escribir tal programa para la máquina de Post que para cualquier n, siendo aplicado al estado arbitrario de la clase D_n, da la parada de resultados en cierto estado de la clase E_{n+1}.

El problema 4 puede ser reducido a los 2 y 3, es decir, se puede indicar el procedimiento de cómo, partiendo de las soluciones arbitrarias de los problemas 2 y 3, se obtiene cierta solución del problema 4. Indiquemos tal procedimiento. Sea Ξ la lista arbitraria de las instrucciones de la máquina de Post, mientras que k, un número entero arbitrario. Designaremos por Ξ $[+k]$ la lista obtenida de Ξ mediante la adición del número k a todos los números y a todos los saltos en las instrucciones de Ξ. Por ejemplo, $\mathrm{I}_1 [+7]$ es semejante lista:

8. $\Leftarrow$ 9
9. $\vee$ 10
10. stop.

Ahora, sea II un programa arbitrario, que es la solución del problema 2, en tanto que III, un programa arbitrario, que es la solución del problema 3. Sea l la longitud del programa II. Ahora, compongamos la siguiente lista de las instrucciones:

$$1.\ ?\begin{matrix} \nearrow l+2 \\ \searrow 2 \end{matrix} \qquad \begin{matrix} \mathrm{II}\,[+1] \\ \mathrm{III}\,[+l+1] \end{matrix}$$

Es fácil verificar que esta lista de instrucciones es el programa para la máquina de Post, con la particularidad de que es un programa que satisface las condiciones del problema 4. En efecto, cualquier estado de la clase D_n pertenece ora a B_n, ora a C_n. En el primer caso «se pone en funcionamiento» la lista II $[+1]$ y en el segundo, la lista III $[+l+1]$.

Sin embargo, el procedimiento expuesto no está obligado a presentar (y, como lo veremos ahora, en realidad no presenta) la solución más corta del problema 4, incluso, partiendo de las soluciones más cortas de los problemas 2 y 3. Veamos qué se obtendrá, si aplicamos este procedimiento a los programas II_1 y III. Obtendremos semejante solución del problema 4:

Programa IV^{10}

1. ? ⟨ 6 ; 2	6. ⇒ 7
2. ⇐ 3	7. ? ⟨ 6 ; 8
3. ? ⟨ 4 ; 2	8. ⇐ 9
4. V 5	9. V 10
5. stop	10. stop.

Está claro, que este programa puede ser reducido, sin cambiar con ello el trabajo que éste realiza, uniendo dos instrucciones de parada en una sola, o bien, como diremos, sobreponiendo una de estas instrucciones en la otra. Es más fácil sobreponer la instrucción N° 5 en la N° 10. Para ello, es suficiente sustituir el salto 10 en todas las instrucciones, donde éste se encuentra (en nuestro caso sólo en la instrucción N° 9) por el salto 5 y, luego, tachar la instrucción N° 10[1]).

Obtendremos el programa IV^9, en el cual ahora se puede sobreponer la instrucción N° 4 en la N° 9. Para ello, es suficiente sustituir el salto 9 en la instrucción N 8 por el salto 4, y, luego, tachar la instrucción N° 9. Después de las dos superposiciones efectuadas obtendremos una nueva solución del problema 4 en forma del programa IV^8.

---

[1]) Si quisiésemos sobreponer la instrucción N° 10 en la N° 5 hubiéramos tenido que sustituir el salto 5 por el 10 en todas las instrucciones, donde éste se encuentra (en el caso dado en la instrucción N° 4), luego, tachar la instrucción N° 5, después de lo que disminuir en 1 todos los números de las instrucciones, que siguen la tachada, y todos los saltos, que coinciden con estos números..

Programa IV[8]

1. ? $\begin{smallmatrix}\nearrow 6\\ \searrow 2\end{smallmatrix}$

2. ⇐ 3

3. ? $\begin{smallmatrix}\nearrow 4\\ \searrow 2\end{smallmatrix}$

4. V 5

5. stop

6. ⇒ 7

7. ? $\begin{smallmatrix}\nearrow 7\\ \searrow 6\end{smallmatrix}$

8. ⇐ 4

Observación. En el caso general la superposición (en dicho programa) de la instrucción N° β en la instrucción N° α consiste en el cumplimiento consecutivo de tres operaciones. En primer lugar, por doquier el salto α se cambia por el β; en segundo lugar, la instrucción N° α se tacha; en tercer lugar, en todas las instrucciones de la lista creada los números $\alpha + 1$, $\alpha + 2$, $\alpha + 3$, . . . (que pueden ser tanto números, como saltos) se cambian por α, $\alpha + 1$, $\alpha + 2$, . . ., respectivamente. Si las instrucciones N° α y N° β coincidieron en todo, salvo sus números, entonces el programa creado, después de sobreponer la instrucción β en la instrucción α, y el programa inicial van a «funcionar del mismo modo en absoluto». Las palabras puestas en la frase anterior entre comillas se precisan del modo siguiente. Tomemos dos ejemplares de la máquina de Post, pongamos cada uno de éstos en cierto—el mismo para ambas máquinas—estado inicial y obliguemos a trabajar la primera máquina según cierto programa A, mientras que la segunda, siguiendo otro programa B. Suponiendo que las máquinas funcionan sincrónicamente, compararemos los estados de la primera y de la segunda máquina que surgen simultáneamente. En el momento inicial estos estados coinciden según el planteamiento. Puede ocurrir que éstos también coinciden en todos los momentos ulteriores, con la particularidad de que la parada de cada máquina sucede—si esto tiene lugar en general—simultáneamente con la parada de la otra máquina y posee la misma cualidad (es decir, ora esta parada es en ambas máquinas de resultados, ora en ambas máquinas, sin resultados). Si el caso descrito se efectúa para cualquier estado inicial, que es general para ambas máquinas, entonces es cuando diremos que los programas A y B trabajan de modo en absoluto

igual. En términos generales, A y B funcionan absolutamente igual, si para cualquier m al transcurrir m pasos del trabajo del programa A surge el mismo estado que después de m pasos del trabajo del programa B, a condición de que esto fue justo para $m = 0$, es decir, en el mismo comienzo del funcionamiento. He aquí ejemplos de los programas que trabajan de modo igual: a) I_1 y I_2; b) I'_1 y I'_2; c) IV^{10} y IV^8.

Ahora veamos si se puede o no reducir también el programa IV^8. En éste no hay dos instrucciones que se distingan sólo por los números, por este motivo el procedimiento de superposición aquí puede llevar, hablando en general, al programa que no trabajará de modo absolutamente igual que el inicial. Pero a pesar de todo, el programa IV^8 también puede ser reducido recurriendo al método de superposición.

Con este fin, compararemosl as instrucciones N° 8 y N° 2. Éstas no pueden ser reunidas en una sola, ya que tienen diversos saltos. No obstante, resulta que la instrucción N° 2 puede, en cierto sentido, tomar en sí funciones de la instrucción N° 8 (y, por esta razón, asimilar la última).

En efecto, sea que en cierta etapa del trabajo del programa tengamos que cumplir la instrucción N° 8. Supongamos que la célula, dispuesta directamente a la izquierda de la que se observa, en el momento examinado, por el carro, está vacía. En este caso, la instrucción N° 8 desplazará el carro a esta célula vacía, después de lo cual la instrucción N° 4 la marcará. Si en lugar de la instrucción N° 8 empezamos por cumplir la instrucción N° 2, sucederá lo siguiente: la instrucción N° 2 desplazará el carro a aquella misma célula vacía, la instrucción N° 3 obligará a la máquina a realizar un «paso sin moverse», después de lo que de nuevo se pondrá en funcionamiento la instrucción N° 4. Veamos qué sucederá si la célula, dispuesta directamente a la izquierda de la observada, no está vacía, sino marcada. Si es así, en el primer caso (es decir, después del cumplimiento de la instrucción N° 8) transcurrirá la parada sin resultados, mientras que en el segundo caso (es decir, una vez ejecutada la instrucción N° 2) la misma no tendrá lugar.

El razonamiento realizado muestra que si el cumplimiento sucesivo de las instrucciones N° 8 y N° 4 no lleva a la parada sin resultados, éste puede ser cambiado por la ejecución de las instrucciones N° N° 2, 3 y 4 (mientras que el cambio recíproco puede, hablando en general, transformar sustancialmente el

trabajo del programa: el cumplimiento de las instrucciones N° N° 2, 3 y 4 nunca lleva a la parada sin resultados, pero al cambiar esta ejecución por el cumplimiento de las instrucciones N° N° 8 y 4 puede surgir la parada sin resultados). Por esta razón, si nos limitamos a tales estados iniciales para los cuales al cumplir el programa IV^8 es imposible la parada sin resultados (entre ellos figuran a ciencia cierta todos los estados de D_n, donde $n = 0, 1, 2, \ldots$ [1]), entonces el programa IV^7 obtenido al sobreponer la instrucción N° 2 en la instrucción N° 8 también será la solución del problema 4. He aquí este programa:

Programa IV^7

1. $?\langle^{6}_{2}$	3. $?\langle^{4}_{2}$	5. stop	7. $?\langle^{6}_{2}$.
2. $\Leftarrow 3$	4. $\vee 5$	6. $\Rightarrow 7$	

Ejercicio. ¿Trabajarán de modo absolutamente igual o no los programas IV^8 y IV^7?

Al efectuar en el programa IV^7 la asimilación de la instrucción N° 7 por la N° 1 obtendremos el siguiente programa IV^6:

Programa IV^6

1. $?\langle^{6}_{2}$	3. $?\langle^{4}_{2}$	5. stop
2. $\Leftarrow 3$	4. $\vee 5$	6. $\Rightarrow 7$.

Puesto que los programas IV^7 y IV^6 funcionan de modo absolutamente igual, entonces el programa IV^6, así como IV^7, será la solución del problema 4. Intenten demostrar que el problema 4 no tiene soluciones más cortas.

De modo completamente análogo se soluciona el problema sobre la adición de la unidad para los estados iniciales, en los cuales la célula observada ora está marcada, ora se encuentra a la derecha del juego marcado. La solución del correspondiente problema —lo llamaremos problema 4′— puede ser obtenida,

[1]) Efectivamente, el programa IV^8 siendo aplicado a estos estados lleva a la parada de resultados.

cambiando en la solución arbitraria del problema 4 todos los signos $\Rightarrow$ por $\Leftarrow$, mientras que los signos $\Leftarrow$ por $\Rightarrow$.

Si designamos por D'_n el conjunto de todos aquellos estados de E_n, en los cuales el carro se encuentra ya sea en el juego, o bien a la derecha de éste, se puede notar que—para cada n—la clase E_n es la unión de las clases D'_n y D_n, mientras que la clase B_n, la intersección de estas clases. En símbolos se anota así:

$$D_n \cup D'_n = E_n, \quad D_n \cap D'_n = B_n.$$

§ 5. ADICIÓN DE LA UNIDAD EN EL CASO MÁS GENERAL

Ahora no pondremos ningunas limitaciones sobre la disposición recíproca del carro y del registro de máquina del número en el estado inicial. Por consiguiente, se exige elaborar el programa que realice la adición de la unidad a condición de que el registro del número inicial se encuentra en una sección arbitraria de la cinta y el carro está parado por doquier. El requisito respectivo se enuncia en el problema 5.

Problema 5. Escribir tal programa para la máquina de Post que para cualquier n, siendo aplicado al estado arbitrario de la clase E_n, da la parada de resultados en cierto estado de la clase E_{n+1}.

Es evidente, que cualquier programa que es la solución del problema 5, servirá al mismo tiempo como solución de cada uno de los problemas anteriores. Sin embargo, ni una sola de las soluciones expuestas antes de los problemas 1, 1′, 2, 3, 3′, 4 y 4′ son soluciones del problema 5 (verifíquenlo).

Los estados iniciales para el problema 5 se ofrecen en las figuras 18, 19 y 20: efectivamente, cualquier estado de E_n pertenece ora a B_n, ora a C_n, ora a C'_n. Sin embargo, ahora hay que elaborar un programa unificado que conduzca al objetivo para cada uno de los aspectos del estado inicial mostrados en las figuras 18, 19 y 20. Intenten hallar semejante programa por su cuenta. Verán que esto no es tan fácil. ¿Puede ser que el programa requerido no exista en absoluto? En este caso intenten demostrar que éste no existe. La respuesta a la cuestión de si tiene o no el problema 5 solución será dada en el siguiente capítulo.

CAPÍTULO III. ANÁLISIS Y SÍNTESIS DE LOS PROGRAMAS PARA LA MÁQUINA DE POST

En los capítulos anteriores ya hemos tropezado tanto con el problema de síntesis de los programas (que consiste en la elaboración de los programas que realizan operaciones prefijadas), como también con el problema de análisis (que consiste en la descripción de aquellas transformaciones que efectúan ros programas). En el presente capítulo estos problemas se resolverán para situaciones más complicadas e interesantes: el análisis de los programas se examinará con motivo de un problema que quedó sin resolver en el capítulo anterior, mientras que la síntesis, en relación al problema acerca de la adición de los números.

§ 1. DIAGRAMAS Y ESQUEMAS SINÓPTICOS

Los programas con los que tropezamos en los dos capítulos anteriores, dedicados a la máquina de Post, eran sencillos.

Por esta razón, pudimos sin dificultad tanto **analizar** el programa prefijado, es decir, comprender como éste trabaja, como también **sintetizar** el programa necesario, es decir, componerlo con las propiedades requeridas.

Pero, supongamos que está dado un programa un poco más complicado, por ejemplo, el siguiente:

Programa V

1. $? \genfrac{}{}{0pt}{}{\nearrow 2}{\searrow 3}$	4. $? \genfrac{}{}{0pt}{}{\nearrow 5}{\searrow 3}$	7. $? \genfrac{}{}{0pt}{}{\nearrow 8}{\searrow 15}$
2. $\Leftarrow 4$	5. $\vee\ 6$	8. $\Rightarrow 9$
3. $\Rightarrow 4$	6. $\Leftarrow 7$	9. $? \genfrac{}{}{0pt}{}{\nearrow 10}{\searrow 8}$

10. $\vee$ 11	14. $?\genfrac{}{}{0pt}{}{5}{13}$	18. ξ 16	21. $?\genfrac{}{}{0pt}{}{23}{22}$
11. $\Rightarrow$ 12	15. $\Rightarrow$ 16	19. $\Leftarrow$ 20	22. ξ 20
12. $?\genfrac{}{}{0pt}{}{13}{19}$	16. $\Rightarrow$ 17	20. $\Leftarrow$ 21	23. slop.
13. $\Leftarrow$ 14	17. $?\genfrac{}{}{0pt}{}{23}{18}$		

¿Cómo analizar este programa?

Procedamos así: dividiremos nuestro programa en partes (que llamaremos *bloques*) e intentaremos llegar a comprender cómo funciona cada bloque por separado y cómo los bloques independientes interaccionan entre sí. Para hallar una división cómoda del programa V en bloques, compongamos el llamado *diagrama* de este programa.

En el diagrama del programa las instrucciones se representan por círculos, mientras que las transiciones de una instrucción a otra, por saetas. Para trazar el diagrama del programa V, dibujaremos 23 círculos, anotándolos con los números de 1 a 23. Tracemos desde el círculo N° i hacia el círculo N° j una saeta si, y sólo si, la i-ésima instrucción (es decir, la del número i) tiene salto igual a j. En particular, si la i-ésima instrucción es la de parada, entonces del i-ésimo círculo no saldrá saeta alguna, mientras que si la i-ésima instrucción es la de salto de control, entonces desde el i-ésimo círculo saldrán dos saetas: una hacia el círculo, que corresponde al salto superior de la instrucción, y la segunda hacia el que corresponde al salto inferior. En este caso, todo nuestro programa se representará por la figura que se ofrece en la fig. 21. Denominaremos esta figura *diagrama* del programa V y los círculos que lo componen, *nudos* del diagrama.

Mediante el procedimiento expuesto puede trazarse el diagrama para cualquier programa. El cambio consecutivo de las instrucciones a ejecutar en dicho programa se ilustra

con evidencia avanzando por el diagrama de este programa en la dirección indicada por las saetas.

Ahora, dividiremos nuestro programa V en grupos de instrucciones, es decir, bloques. Por esta misma razón, en bloques (grupos de nudos) también será dividido el programa. Es más evidente prefijar la división en bloques precisamente como división del diagrama.

Para nuestras finalidades de análisis del programa V es más cómodo dividir el diagrama mostrado en la fig. 21 y junto con

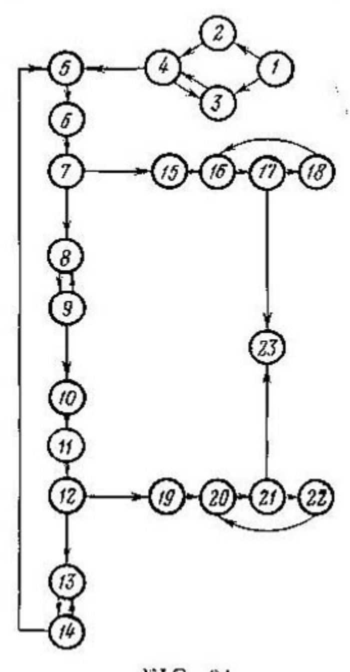


FIG. 21

éste también el propio programa V en los siguientes ocho bloques:

1-er bloque, lo llamaremos «bloque del comienzo», consta de los nudos (instrucciones) N°N° 1, 2, 3 y 4;

2-do bloque, lo llamaremos «bloque de comprobación a la

izquierda», consta de los nudos (instrucciones) N°N° 5, 6 y 7:

3-er bloque, lo llamaremos «bloque de movimiento a la derecha», consta de los nudos (instrucciones) N°N° 8 y 9;

4-to bloque, lo llamaremos «bloque de comprobación a la derecha», consta de los nudos (instrucciones) N°N° 10, 11 y 12;

5-to bloque, lo llamaremos «bloque de movimiento a la izquierda», consta de los nudos (instrucciones) N°N° 13 y 14;

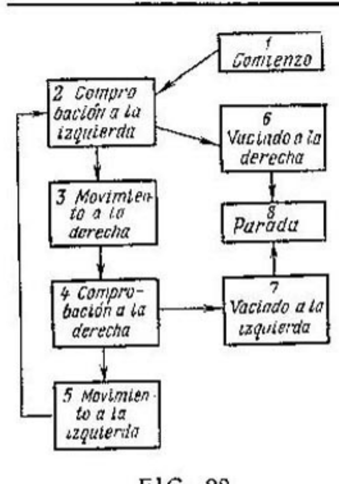


FIG. 22

6-to bloque, lo llamaremos «bloque de vaciado a la derecha», consta de los nudos (instrucciones) N°N° 15, 16, 17 y 18;

7-mo bloque, lo llamaremos «bloque de vaciado a la izquierda», consta de los nudos (instrucciones) N°N° 19, 20, 21 y 22;

8-vo bloque, lo llamaremos «bloque de parada», consta del nudo (instrucción) N° 23.

Para cada división del programa prefijado en bloques puede ser trazado el llamado *esquema sinóptico* (en bloques) de dicho programa. Para ello, es preciso representar cada bloque en forma de un rectángulo y trazar del rectángulo, que representa el bloque α, al que representa el bloque β tal cantidad de saetas, cuantas salen de los nudos del bloque α a los nudos del bloque β. El esquema sinóptico del programa V, para la división en bloques elegida por nosotros, se ofrece en la fig. 22

(dentro de cada rectángulo están inscritos el número y la denominación del bloque correspondiente).

Teniendo ante nosotros el esquema sinóptico de cierto programa, podemos imaginarnos con evidencia su funcionamiento en forma de la ejecución consecutiva de las instrucciones de uno u otro bloque. El bloque que contiene la instrucción N° 1 lo denominaremos *inicial o de comienzo*, mientras que el bloque que contiene la instrucción de parada, *final*; a fin de simplificar consideraremos que el bloque final no contiene nada más. En vez de mencionar la «ejecución de las instrucciones del bloque prefijado» convengamos decir, para brevedad, simplemente «ejecución del bloque prefijado». En primer lugar siempre se ejecutará el bloque de comienzo.

Durante la ejecución de cierto bloque, que no sea el final, pueden tener lugar tres casos: 1) en el tiempo de ejecución del bloque sucederá una parada sin resultados; 2) la ejecución del bloque nunca terminará, es decir, cada vez después de cumplir cierta instrucción habremos de ejecutar de nuevo la instrucción del mismo bloque; 3) el cumplimiento del bloque culminará, es decir, una vez ejecutada cierta instrucción habremos de pasar al cumplimiento de alguna instrucción de otro bloque. Hablando brevemente, al caer en el bloque que no es final, ora nos «atascaremos» en éste para siempre, ora por una de las saetas que parten de él llegaremos a otro bloque del sistema.

A veces, se puede en seguida, ateniéndose al aspecto del diagrama destacar algunos bloques, en los cuales a ciencia cierta es imposible atascarse (claro está excluyendo el caso de una parada sin resultados que puede suceder en cualquier momento). Para el esquema sinóptico mostrado en la fig. 22 tales serán, como se infiere del diagrama en la fig. 21, los bloques 2-do y 4-to. Por esta razón, el cumplimiento del programa V se efectuará del modo siguiente: en primer lugar se ejecuta el 1-er bloque; su cumplimiento nunca finaliza, o bien culmina y en este caso se ejecuta el 2-do bloque. La ejecución del 2-do bloque tiene fin sin falta y después de él se ejecuta ora el 3-er bloque, ora el 6-to; detrás del 3-ro (si su cumplimiento culmina) va el 4-to; después del 4-to ora el 5-to, ora el 7-mo; tras del 5-to (si su ejecución culmina) va el 2-do; después del 2-do de nuevo ora el 3-ro, ora el 6-to; etc. El cumplimiento del bloque 6-to (así como también del 7-mo) puede tener fin, pero

también no tenerlo, en caso de que lo tenga, se cumple el 8-vo bloque y la máquina se para.

Ahora, por fin, podemos resolver para nuestro programa el problema del análisis. Con ello, nos limitamos al caso cuando el estado inicial pertenece (para cierto n) a la clase E_n. Con ayuda del esquema sinóptico en el siguiente párrafo se mostrará que aplicando a tal estado inicial el programa lleva a la parada de resultados en el estado de la clase E_{n+1}.

De este mismo modo se resuelve el siguiente problema general acerca de la adición de la unidad (planteado en el párrafo 5 del capítulo anterior):

escribir tal programa para la máquina de Post que para cualquier n, siendo aplicado al estado arbitrario de E_n (tomado en calidad de inicial), da una parada de resultados en cierto estado de la clase E_{n+1}.

Precisamente el programa V de la longitud 23 resulta ser la solución de este problema. El autor no ha conseguido elaborar un programa más corto que sirviera de solución del mismo problema. Además, el autor no sabe cómo demostrar que el programa hallado es el más corto de los posibles. ¿Pueda ser que alguno de los lectores consiguiera hacer lo uno o lo otro?

§ 2. ANÁLISIS DEL PROGRAMA DE ADICIÓN DE LA UNIDAD

Pues bien, pasemos al análisis del programa V del § 1, es decir, a la aclaración de cómo éste funciona. Como hemos acordado, en calidad de estado inicial elegimos cierto estado que pertenece a una de las clases E_n ($n = 0, 1, 2, \ldots$). Designemos por H_n la clase de aquellos estados de la clase E_n, para la cual tanto la célula observada, como su vecina a la derecha están vacías.

Comencemos a ejecutar el programa V. Al principio se cumple el primer bloque. Ahora mostremos que su cumplimiento tiene fin (es decir, tiene salida al segundo bloque); simultáneamente veremos qué estado de la máquina surgirá después de su ejecución. Examinaremos tres casos.

Caso 1. En el estado inicial tanto la célula observada, como la vecina de ésta a la izquierda están vacías. En este caso, se ponen en funcionamiento sucesivamente las instrucciones N°N° 1, 2 y 4, después de lo que la ejecución del bloque culminará, con la particularidad de que la máquina llegará al estado de la clase H_n.

Caso 2. En el estado inicial la célula observada está vacía, pero su vecina izquierda viene marcada. Esto significa que el juego de la

longitud $n + 1$, que es el que nos interesa, se encuentra a la izquierda de la célula observada y, por consiguiente, todas las células a la derecha de ésta están vacías. En este caso, se ponen en funcionamiento de forma consecutiva las instrucciones N°N° 1, 2, 4, 3 y 4, después de lo que culminará el cumplimiento del bloque, con la particularidad de que la máquina llegará al estado de la clase H_n.

Caso 3. En el estado inicial la célula observada está marcada. Esto significa que el carro se encuentra exactamente frente a una de las células de nuestro juego de la longitud $n + 1$; sea que aquél se encuentra en la k-ésima célula a la derecha de este juego. En este caso, funcionarán las instrucciones N°N° 1, 3, 4, 3, 4, 3, 4, . . ., con la particularidad de que la ejecución del par de instrucciones N°N° 3 y 4

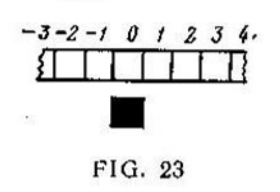


FIG. 23

transcurrirá exactamente k veces hasta que el carro vaya a parar a la primera célula vacía a la derecha; después de esto la ejecución del bloque culminará y la máquina se encontrará en uno de los estados de la clase H_n.

Así pues, para cualquier estado de la clase E_n que se aplique el programa V, la ejecución del primer bloque tendrá fin, después de lo que la máquina se encontrará en uno de los estados de la clase H_n.

Por esta razón, para realizar nuestro objetivo (que consiste en establecer que el programa V es la solución del problema general acerca de la adición de la unidad) es suficiente demostrar que aplicando este programa, *a partir del 2-do bloque*, al estado arbitrario de la clase H_n, obtendremos, tarde o temprano, la parada de resultados en el estado de la clase E_{n+1}. Ahora vamos a demostrar este hecho.

De ese modo, sea que la máquina esté en cierto estado de la clase H_n. Fijemos este estado, es decir, designémoslo por h. Sabemos que en la cinta está introducido el «sistema de coordenadas de números enteros», de acuerdo con el cual las células de la cinta están numeradas con números enteros. Olvidemos este «viejo» sistema de coordenadas e introduzcamos el siguiente, nuevo (también de números enteros): numeremos con el número 0 la célula observada en el estado h; las células que se encuentran a la derecha y a la izquierda de ésta, con los números ± 1, ± 2, ± 3, . . . (fig. 23). Subrayemos que este sistema de coordenadas depende del estado h de H_n examinado por nosotros. En este caso, el juego que sirve como registro del número n se dispone en las células con los números m, $m + 1$, . . ., $m + n$, donde m es cierto número positivo o negativo. Las células con los números 0 y 1 están a todas luces vacías. Por este motivo ora $m + n < 0$, ora $m < 1$.

Designemos por H_n^m aquel estado de la máquina, para el cual vienen marcadas las células con los números m, $m+1$, . . ., $m+n$ y sólo éstas, mientras que el carro se encuentra frente a la célula N° 0. En este caso, la clase de los estados H_n consta de los estados H_n^m, donde ora $m+n<0$, ora $m>1$, es decir, de los estados

a) H_n^{-n-1}, H_n^{-n-2}, H_n^{-n-3}, . . .;
b) H_n^2, H_n^3, H_n^4, . . .

La vista general de estos estados se muestra en la fig. 24. Nuestro estado inicial h es uno de estos estados H_n^m.

Por consiguiente, necesitamos demostrar que bajo la acción del programa, a partir del 2-do bloque, cada uno de los estados H_n^m, donde

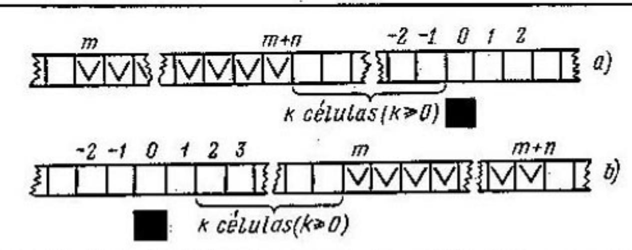


FIG. 24. Estado H_n^m: *a*) para $m+n<0$; *b*) para $m>1$

ora $m+n<0$, ora $m>1$, pasa de la parada de resultados al estado de la clase E_{n+1}. Ahora estableceremos esto, observando qué cambios sufre el estado H_n^m al ejecutar consecutivamente los bloques del programa.

Con este fin introduzcamos varias designaciones. Sea que los números enteros m, n, p y q satisfagan las siguientes tres condiciones: 1) $p \leqslant q$; 2) $n \geqslant 0$; 3) ora $m+n<p$, ora $m>q$. Por $W_n^m\,(p,\,q)$ designemos la clase de todos aquellos estados de la máquina de Post que satisfacen las siguientes dos condiciones: 1) las células con los números p y q están vacías, mientras que todas las células entre ellas vienen marcadas; 2) las células con los números $(m-1)$ y $(m+n+1)$ están vacías, en tanto que todas las células entre ellas están marcadas.

La vista general de los estados de la clase $W_n^m\,(p,\,q)$ viene expuesta en la fig. 25. Designemos por $R_n^m\,(p,\,q)$ aquel estado de la clase $W_n^m\,(p,\,q)$, en el cual el carro se encuentre frente a la célula con el número p, mientras que por $S_n^m\,(p,\,q)$, aquel estado de la clase $W_n^m\,(p,\,q)$ en el cual el carro está frente a la célula con el número q.

Mediante la comprobación directa, pueden establecerse las siguientes afirmaciones:

1ª. $H_n^m = R_n^m\,(0,\ 1)$.

2ª. Si, antes de comenzar el funcionamiento del segundo bloque, el estado de la máquina de Post era $R_n^m\,(x,\,y)$, además, $x-1 \neq m+n$, entonces una vez ejecutado el 2-do bloque el estado de la

máquina será $R_n^m (x-1, y)$ y, a continuación, deberá ejecutarse el 3-er bloque.

3ª. Si, antes del comienzo del funcionamiento del 3-er bloque, el estado de la máquina de Post era $R_n^m (x, y)$, la ejecución de este bloque culminará, después de lo que el estado de la máquina de Post será $S_n^m (x, y)$.

4ª. Si, antes de comenzar el funcionamiento del 4-to bloque, el estado de la máquina de Post era $S_n^m (x, y)$, además, $y+1 \neq m$, entonces después de ejecutar este bloque el estado de la máquina será

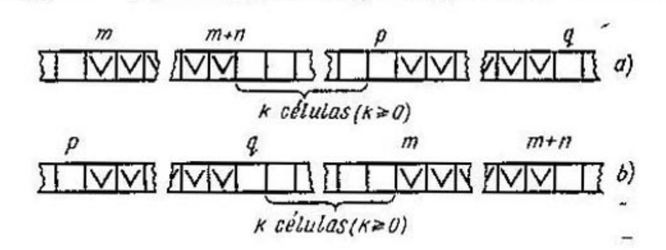


FIG. 25. El estado de la clase $W_n^m (p, q)$ tiene uno de los siguentes aspectos: a) (para $m+n<p$) o bien b) (para $m>q$); el carro puede encontrarse dondequiera

$S_n^m (x, y+1)$ y, a continuación, deberá ejecutarse el quinto bloque.

5ª. Si, antes del comienzo del funcionamiento del 5-to bloque, el estado de la máquina de Post era $S_n^m (x, y)$, la ejecución de este bloque tendrá fin, después de lo que el estado de la máquina será $R_n^m (x, y)$.

Representaremos de manera simbólica la ejecución del bloque por un rectángulo con el número del mismo ubicado en su interior; los estados antecedentes a la ejecución del bloque y los que surgen a resultas de esta ejecución los escribiremos de modo contiguo a la izquierda y a la derecha del rectángulo correspondiente. Entonces, a base de las afirmaciones 1ª . . . 5ª, el trabajo de la máquina de Post, procedente del estado H_n^m y que cumple nuestro programa, a partir del 2-do bloque, se puede representar mediante la siguiente sucesión:

$H_a^m = R_n^m (0, 1)$ [2] $R_n^m (-1, 1)$ [3] $S_n^m (-1, 1)$

[4] $S_n^m (-1, 2)$ [5] $R_n^m (-1, 2)$ [2] $R_n^m (-2, 2)$

[3] $S_n^m (-2, 2)$ [4] $S_n^m (-2, 3)$ [5] $R_n^m (-2, 3)$

[2] $R_n^m (-3, 3)$ . . .

Esta sucesión continúa hasta que llegue a uno de los dos siguientes acontecimientos.

Primer acontecimiento. En cierto instante de tiempo, antes de comenzar el trabajo del 2-do bloque (puede ser, en particular, en el mismo inicio), el estado de la máquina resulta ser tal $R_n^m(x, y)$ que $x - 1 = m + n$.

Segundo acontecimiento. En algún instante de tiempo, antes de que comience el funcionamiento del 4-to bloque, el estado de la máquina resulta ser tal $S_n^m(x, y)$ que $y + 1 = m$.

Notemos, que uno de estos acontecimientos ocurrirá sin falta (pregunta al lector: ¿porqué?).

Examinemos cada uno de estos acontecimientos por separado.

Primer acontecimiento. Sea que cuando comenzó a funcionar el 2-do bloque el estado de la máquina es $R_n^m(x, y)$, donde $x - 1 =$

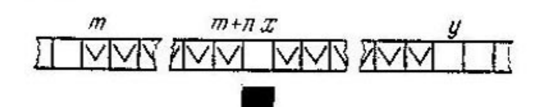


FIG. 26. Aquí se muestra el estado R_n^m (para $m + n + 1$, y)

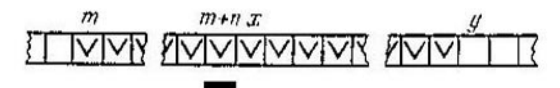


FIG. 27. He aquí el estado que surge, si el 2º bloque se aplica al estado que viene representado en la fig. 26

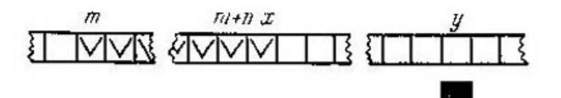


FIG. 28. He aquí el estado que surge una vez cumplido el 6º bloque, aplicado al estado expuesto en la fig. 27. A continuación, en este estado se realizará una parada de resultados

$= m + n$. Este estado se ofrece en la fig. 26. Una vez ejecutado el 2-do bloque surge el estado mostrado en la fig. 27. La útlima instrucción del 2-do bloque —la instrucción Nº 7— «transmite el control» al 6-to bloque, es decir, precisamente el 6-to bloque deberá ejecutarse ahora. Es fácil de verificar que el cumplimiento del 6-to bloque culminará y después de esto surgirá el estado ofrecido en la fig. 28. A este estado se aplicará el 8-vo bloque que llevará a la parada de resultados en esta posición.

Segundo acontecimiento. Sea que para el comienzo del funcionamiento del 4-to bloque el estado de la máquina es $S_n^m(x, y)$, donde $y + 1 = m$. Este estado viene expuesto en la fig. 29. Es fácil de comprobar que una vez culminado el 4-to bloque se ejecutará el 7-mo y, luego, el 8-vo, que llevará a la parada de resultados en el estado ofrecido en la fig. 30.

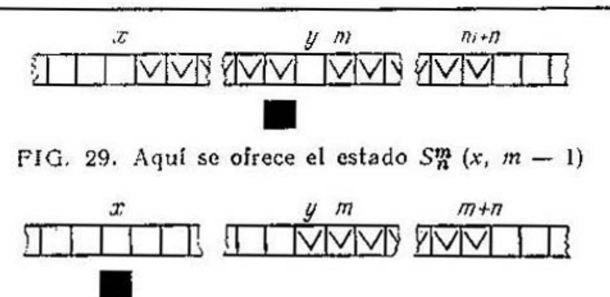


FIG. 29. Aquí se ofrece el estado $S_n^m(x, m - 1)$

FIG. 30. En este estado ocurrirá una parada de resultados después de ejecutar los bloques 4º, 7º y 8º (conforme al estado que se muestra en la fig. 29)

Nos queda notar que tanto el estado de la fig. 28, como también el de la fig. 30 pertenecen a la clase E_{n+1}. Por lo tanto, en cualquiera de los dos posibles casos el trabajo de la máquina terminará por una parada de resultados con el surgimiento del estado de la clase E_{n+1}, lo que se pretendía demostrar.

§ 3. UNA VEZ MÁS SOBRE EL PROBLEMA DE ADICIÓN DE LA UNIDAD

Así pues, el problema general acerca de la adición de la unidad está resuelto. Ahora, analizaremos su solución. Desde luego, el lector estará de acuerdo con que la dificultad fundamental, que surge al revolver este problema, no consiste en aumentar en una marca el juego de la longitud $n + 1$, registrado en la cinta, sino que en hallar este juego. Durante la resolución de los problemas más simples acerca de la adición de la unidad, examinados en el capítulo anterior, sabíamos en qué dirección hay que buscar el juego y, correspondientemente, escribíamos el programa, cuyo funcionamiento comenzaba por el movimiento hacia el lado respectivo (si no resultaba que el carro ya al inicio se encontraba en el juego buscado). En el problema general no sabemos hacia dónde tenemos que movernos. Es posible, que precisamente por esto a ciertos lectores este problema les pareció al principio insoluble.

¿De qué modo se puede «llegar» a que se precisa el programa V u otro análogo, que probablemente el lector compondrá o bien ya ha hallado por su propia iniciativa? He aquí cómo. Imaginémonos que estamos en un camino infinito por ambos lados y en algún sitio de este camino yace una piedra maravillosa que deseamos encontrar. No sabemos dónde ésta yace, sabemos sólo que sin falta está en algún lugar. ¿Cómo obrar? Al elegir cualquiera de los lados puede suceder que la piedra maravillosa yace, precisamente, en el lado contrario. Es evidente, que debemos alternar el movimiento de un lado al otro, aumentando constantemente la amplitud de las oscilaciones: primero damos un paso a un lado, luego, dos pasos al otro, después, de nuevo tres pasos al primer lado, luego, cuatro pasos al segundo, etc., hasta que tropecemos con la piedra maravillosa.

Ahora reflexionemos sobre lo siguiente. ¿Cómo determinar el momento cuando hay que dar la vuelta, es decir, cambiar la dirección de movimiento? Puede parecer fácil: sólo se debe efectuar la cuenta de los pasos y saber en cada instante cuántos pasos debemos realizar en dicha dirección y cuántos pasos de esta cantidad ya hemos dado. Pero en la realidad, existe aquí una dificultad consistente en cómo fijar los números de pasos que necesitamos. Si almacenamos éstos en nuestra memoria, debemos estar listos para recordar números cuan se quiera grandes, lo que es imposible: efectivamente, si la piedra maravillosa yace bastante lejos de nuestra posición inicial, entonces tendremos que recordar números que superan las posibilidades de nuestro cerebro. Se pueden tomar notas de la cantidad de pasos, digamos, en libretas de apuntes llevadas consigo, pero, en este caso, tendremos que estar dispuestos a llevar notas cuan se quiera voluminosas (además, la lectura de estas notas será un problema independiente). Si prohibimos que se lleven consigo cualesquier apuntes, la dificultad indicada podrá parecer, a primera vista, insuperable. Sin embargo, hay la siguiente salida de esta dificultad: se puede utilizar para los apuntes el propio camino. Obraremos así. Precisamente imitando a Pulgarcito marcaremos nuestra ruta a lo largo del camino con migajas o piedrecitas: al hacer un paso colocaremos ante sí, por ejemplo, una piedrecita (claro está si, ésta ya no se encuentra allí). Por lo tanto, andando tras las piedras y llegando hasta su fin, colocaremos una nueva piedrecita y daremos la vuelta al lado opuesto —y así hasta que encontremos la piedra maravillosa. He aquí el método de búsqueda que está realizado en el programa V; el papel de piedrecitas lo juegan aquí las marcas inscritas por el carro, para marcar las células que pasó. Cabe señalar, además, que en nuestro caso el papel de piedra maravillosa lo juega también la marca, por esta razón es necesario tomar medidas especiales, para no confundir la «piedrecita» con la «piedra maravillosa» (en el programa V estas medidas fueron realizados por los bloques de comprobación). La «insuficiencia» del procedimiento de Pulgarcito consiste en que es necesario tener en el bolsillo una reserva ilimitada de piedrecitas o migajas (sin ninguna duda, desde el punto de vista de la máquina de Post esta insuficiencia es de poca importancia). Por lo demás, nos serán suficientes tan sólo dos piedrecitas. Para ello, es preciso, primero, colocar una piedrecita a la izquierda de sí y la otra, a la derecha, luego, andar entre ellas trasladándolas; a saber-

cada vez al tropezar con una piedrecita hay que trasladarla a un paso adelante, dar la vuelta y caminar hasta la otra piedrecita, con la cual hacer lo mismo.

Ejercicio. Efectúen mediante el programa para la máquina de Post el método recién expuesto de búsqueda de la piedra maravillosa con ayuda de dos piedrecitas. Comparen la longitud del programa obtenido con la longitud del programa V.

Volvamos por última vez al examen del programa V. El papel de cada bloque en él viene definido por su denominación: los bloques de movimiento desplazan el carro a lo largo del juego de las marcas inscritas hasta llegar al fin de éste, los bloques de comprobación verifican si el carro ya ha llegado o no cerca del registro buscado del número, los bloques de vaciado borran todas aquellas marcas, con las que el carro marcaba su ruta. Si fijamos al principio el estado de la clase H_n y realizamos el programa comenzando desde el 2-do bloque, se obtendrá el resultado requerido. (Por lo tanto, si exigimos que el programa dé resultado sólo con arreglo al estado de H_n, podríamos limitarnos a las últimas 19 instrucciones, claro está, disminuyendo con anticipación todas las direcciones y los saltos en 4.) El objetivo del bloque de comienzo consiste en hacer pasar la máquina del estado arbitrario de la clase E_n a cualesquiera de los estados de la clase H_n. Notemos que el bloque de comienzo trabaja en todos los casos: incluso si el carro en el estado inicial ya se encuentra en el registro buscado del número, el bloque de comienzo desplazará el carro de este juego para que, luego, empiecen las búsquedas de este mismo juego. Es evidente, que esto parece poco razonable. ¿No sería más fácil, primero identificar si al comienzo el carro se encuentra o no en el juego (es decir, tiene lugar o no el caso 3 del párrafo anterior), si se encuentra, entonces añadir una marca (como lo exige el problema 2 del párrafo 3 del capítulo anterior); si no, desplazarse a una célula a la izquierda y observar si ésta posee marca (o sea, tiene lugar o no el caso 2 del mencionado párrafo); en caso de que esto sea así, añadir al juego hallado una marca (como lo requiere el problema 1′ del párrafo 2 del capítulo anterior); si esto no es así y únicamente en este caso (es decir en el caso 1 del párrafo anterior), con dos «no» deberán incluirse las instrucciones N°N° 5...23 del programa V?

Compongamos el programa que realice este plan. Es cómodo elaborar el programa junto con el correspondiente esquema sinóptico. Para ello, designemos por II cualquiera de los programas de la longitud 4 que sirven como solución del problema 2 del capítulo anterior, por I′ cualquiera de los programas de la longitud 3 que sirven en calidad de solución del problema 1′ del mismo capítulo y por Γ, la lista de instrucciones del N° 5 al N° 23 del programa V. El programa buscado se muestra mediante su esquema sinóptico en la fig. 31. Recordemos que de acuerdo con la designación del párrafo 4 del capítulo anterior para cualquiera de las listas de instrucciones Ξ por $\Xi\,[+\,m]$ se designa la lista que se obtiene de Ξ mediante el aumento en m de todos los números de instrucciones y de todos los saltos.

El programa en la fig. 31, eligiendo de modo oportuno el programa II (a saber, en caso de que el programa II contenga la instrucción de movimiento a la derecha, es decir, represente de por sí uno de los

programas $\Pi'_1, \Pi'_2, \ldots, \Pi'_{12}$ mencionados en el ejercicio 3 del párrafo 3 en el capítulo anterior), conduce al resultado recorriendo un camino no más largo que en el programa V, es decir, para todo estado inicial exige no más pasos de trabajo de la máquina que el programa V (aconsejamos al lector que compruebe que esto no es así, cuando el programa Π contiene la instrucción de movimiento a la izquierda, es decir, representa de por sí uno de los programas $\Pi_1, \Pi_2, \ldots, \Pi_{12}$). En algunos casos el programa ofrecido en la figura 31 lleva a resultados incluso por un camino más corto que el programa V, es decir, exige

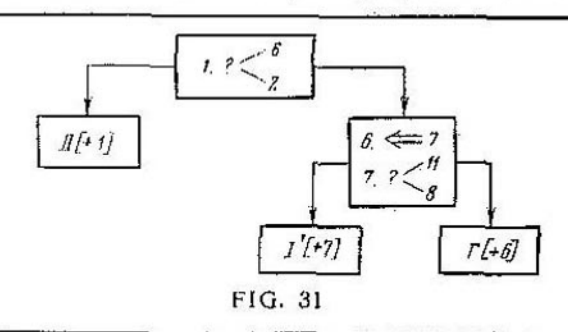


FIG. 31

menor número de pasos de trabajo de la máquina. Proponemos al lector que compruebe esto y se cerciore de que el programa en la fig. 31 exige, para obtener una parada de resultados, ora el mismo número de pasos que el programa V (para los estados iniciales que corresponden al caso 1 del párrafo anterior), ora 6 pasos menos (para los estados iniciales que corresponden al caso 2), ora 5 pasos menos (para los estados iniciales que corresponden al caso 3).

En cambio, el programa mostrado en la fig. 31 tiene 29 instrucciones. Incluso si se juntan en una sola tres de las instrucciones de parada que hay en este programa, a pesar de todo quedan 27 instrucciones, es decir, más de 23. Esto no es asombroso, ya que aquí, en realidad, hay tres distintos programas para cada uno de los tres casos posibles indicados en el párrafo anterior, que pueden revelarse en el comienzo. El papel del juego del «bloque de comienzo», que es a primera vista un poco paradójico, en el programa V consiste, precisamente, en la reducción de todos los casos posibles a uno, es decir, al estado de la clase H_n. A costa de esto se obtiene también economía en la longitud del programa. Por lo tanto, aquí hacemos uso de uno de los procedimientos estándares de las matemáticas, es decir, el método que consiste en el intento de reducir la solución de cualesquiera de los problemas planteados a la solución de los que ya están resueltos. (En el ejemplo examinado reducimos la solución del problema, acerca de la adición de la unidad para el estado inicial arbitrario, a la solución de este mismo problema para los estados de H_n.) Por regla, el procedimiento estándar indicado lleva, al ser realizado con éxito, a economía de diversos tipos, por ejemplo, a la del volumen de la memoria necesario

para almacenar el programa (y en el caso general—el método de resolución) de una u otra clase de problemas. Por consiguiente, dicho procedimiento juega en las matemáticas un papel muy importante. No en vano, en el folklore matemático existe la siguiente anécdota, de la cual los matemáticos (por lo menos algunos de ellos) incluso están un poco orgullosos; ésta consiste en el conjunto de dos problemas.

Primer problema. Sean dados: unas cerillas, un infiernillo apagado, un grifo cerrado, una tetera vacía; se exige hervir el agua. **Solución.** Encendamos una cerilla y con ella el infiernillo, abramos el grifo, llenemos la tetera, pongámosla en el infiernillo e hirvamos el agua.

Segundo problema. Sean dados: un infiernillo encendido, un grifo abierto, una tetera llena de agua; se exige hervir el agua. **Solución.** Reduzcamos el problema al que ya está resuelto (es decir, al primero). Con este fin apaguemos el infiernillo, vaciemos la tetera, cerremos el grifo, después de lo cual queda sólo ejecutar las operaciones indicadas en la solución del primer problema.

Observación. La cantidad de operaciones que se requieren para reducir el segundo problema al primero puede ser abreviada, haciendo coincidir el apagamiento del infiernillo con el vaciado de la tetera: es suficiente verter el agua sobre el infiernillo.

§ 4. ADICIÓN DE LOS NUMEROS EN CASOS SIMPLES

Ahora nos ocuparemos de la adición de números en la máquina de Post. En calidad de sumandos se examinarán sólo números no negativos enteros. Como antes, representaremos el número no negativo entero m por un juego de la longitud $m+1$. Efectuar en la máquina de Post la adición de números significa elaborar tal programa que al ser aplicado a la cinta que contiene registros de los números $m_1, m_2, \ldots, m_k$ $(k \geqslant 2)$ conduzca a una parada de resultados, con la particularidad de que después de esta parada en la cinta resulte el registro del número $m_1 + m_2 + \ldots + m_k$. Es necesario hacer una serie de precisiones. Ante todo, siempre supondremos que al principio en la cinta no viene registrado nada, salvo los números $m_1, \ldots, m_k$, y exigir que al final quede registrada tan sólo su suma; no pondremos ninguna restricción en la posición del carro al final, pero, en lo que se refiere a su posición en el comienzo, para simplificar, consideraremos que en el estado inicial el carro se encuentra frente a la célula más izquierda entre las marcadas.

A continuación, pueden ser hechas varias suposiciones sobre la distancia entre los juegos que sirven para registrar los números. (Se suele denominar distancia entre dos juegos el nú-

mero de células, comprendidas entre la célula derecha extrema del juego izquierdo y la izquierda extrema del juego derecho.) En el caso más simple (por el cual comenzamos) la distancia entre los juegos lindantes es igual a 1. Por último, se pueden superponer unas u otras restricciones en la cantidad de sumandos, considerando esta cantidad como prefijada con antelación o arbitraria.

El caso más simple se examina en el problema «a».

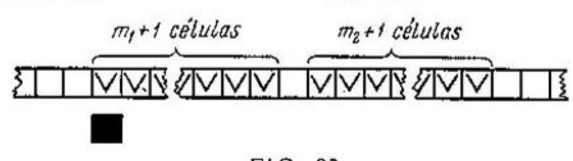


FIG. 32.

Problema «a». Elaborar el programa de adición de dos números registrados a distancia 1 uno del otro.

En virtud de las suposiciones hechas, el estado inicial para el problema «a» tendrá el aspecto expuesto en la figura 32.

Lo más fácil es resolver este problema «llenando» con la marca la célula vacía entre los juegos. En este caso, si los sumandos son los números m_1 y m_2, en la cinta surgirán $m_1 + m_2 + 3$ de células marcadas, mientras que éstas deben ser $m_1 + m_2 + 1$. Por este motivo hay que vaciar las dos marcas sobrantes. En el programa $\tilde{A}$, escrito a continuación, se realiza el plan enunciado de operaciones.

Programa $\tilde{A}$

1. $\Rightarrow$ 2	4. $\Rightarrow$ 5	7. § 8
2. ? 3 / 1	5. ? 6 / 4	8. $\Leftarrow$ 9
		9. § 10
3. V 4	6. $\Leftarrow$ 7	10. stop.

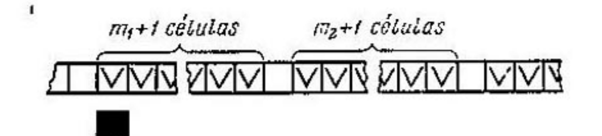
Obtendremos economía en el número de instrucciones si vaciamos dos marcas no en el final, sino en el comienzo, directamente del juego izquierdo. Con ello, hay que tener

precaución para considerar el caso en que el juego izquierdo contiene una sola marca. Este plan se realiza en el programa A que contiene 8 instrucciones en totalidad. La 3-ra instrucción de este programa, precisamente, considera el caso en que en el juego izquierdo hay tan sólo una marca.

Programa A

1. § 2	4. § 5	7. V 8
2. ⇒ 3	5. ⇒ 6	8. stop.
3. ? ↗8 ↘4	6. ? ↗7 ↘5	

Problema «b». Componer el programa de adición de una cantidad arbitraria de números registrados en la cinta a distancia 1 uno del otro.



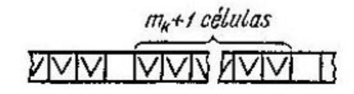


FIG. 33

En virtud de las suposiciones hechas, el estado inicial para el problema «b» será el ilustrado en la fig. 33. Por consiguiente, es necesario elaborar semejante programa que, para cualquier número k y cualesquier números $m_1, m_2, \ldots, m_k$, dé una parada de resultados en el estado en el cual en la cinta estuviera registrado el número $m_1 + m_2 + \ldots + m_k$. He aquí el programa más corto (con mayor precisión, uno de los más cortos) de semejantes programas que conoce el autor:

Programa B

1. § 2	5. ⇒ 6	9. ? $\genfrac{}{}{0pt}{}{\nearrow 10}{\searrow 8}$
2. ⇒ 3	6. ? $\genfrac{}{}{0pt}{}{\nearrow 7}{\searrow 8}$	10. ⇒ 11
3. ? $\genfrac{}{}{0pt}{}{\nearrow 4}{\searrow 2}$	7. stop	11. § 12
4. V 5	8. ⇐ 9	12. ⇒ 1.

Aconsejamos al lector que compruebe este programa (como también todos los programas restantes del presente párrafo) en cualquier ejemplo concreto y, de esta manera, llegue a

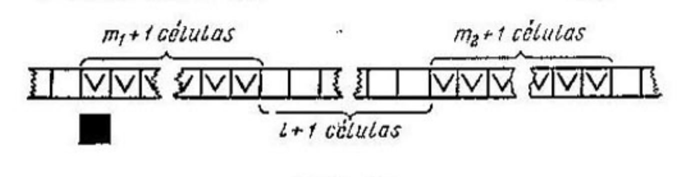


FIG. 34

comprender la «idea» de su funcionamiento. Esta «idea» en el presente caso consiste en que nosotros, de modo sucesivo, cortamos dos marcas a la izquierda (con las instrucciones N° 1 y N° 11), colocando en cambio una marca en la célula vacía próxima a la derecha (mediante la instrucción N° 4); si la célula que sigue a la vacía próxima a la derecha también está vacía (lo que se verifica por medio de las instrucciones N° 5 y N° 6), esto significa que ya es hora de terminar (salto del control a la instrucción N° 7).

Ahora pasemos a la adición de los números registrados en la cinta a una distancia **arbitraria** entre ellos.

Problema «c». Elaborar el programa de adición de dos números registrados a una distancia arbitraria uno del otro.

El estado inicial para el problema «c» se muestra en la fig. 34. Es necesario componer el programa que nos lleve al obje-

tivo, cualesquiera que sean los números no negativos enteros m_1, m_2, l. He aquí uno de tales programas:

Programa $\tilde{C}$

1. ξ 2	6. ? 7 / 10	10. ⇐ 11
2. ⇒ 3	7. ⇐ 8	11. ? 12 / 11
3. ? 4 / 2	8. ? 9 / 7	12. ⇒ 13
4. ∨ 5	9. ⇒ 1	13. ξ 14
5. ⇒ 6		14. stop.

La «idea» de este programa consiste en que el juego izquierdo «se desplaza» a la derecha hasta hacerlo unirse con el derecho. El desplazamiento del juego se efectúa mediante el «traslado» de la marca más a la izquierda del juego a la célula vacía próxima a la derecha (la marca se corta por la instrucción N° 1 y se pone por la instrucción N° 4). Cuando los juegos llegan a unirse (lo que se descubre con las instrucciones N° 5 y N° 6), resultan marcadas las $m_1 + m_2 + 2$ células, es decir, en una más de las que se requieren. Las instrucciones N° N° 10, 11 y 12 trasladan el carro al extremo izquierdo del juego, donde se vacía la marca excesiva (mediante la instrucción N° 13). Por lo tanto (esto es muy importante para lo sucesivo), en el estado final el carro se encuentra directamente a la izquierda de la suma formada.

Al resolver el problema «a» ya hemos visto que se puede reducir la longitud del programa, si se vacía la marca sobrante no al final, sino al comienzo del funcionamiento de la máquina. Al hacer uso de este procedimiento, obtenemos el siguiente programa C_1 de la longitud 12 que realiza el mismo método de desplazamiento del juego izquierdo a la derecha y, por eso, es muy parecido al programa $\tilde{C}$:

Programa C_1

1. ξ 2	5. ⇒ 6	9. ? ↗10 ↘12
2. ⇒ 3	6. ? ↗7 ↘5	10. ⇐ 11
3. ? ↗12 ↘4	7. ∨ 8	11. ? ↗2 ↘10
4. ξ 5	8. ⇒ 9	12. stop.

El autor no conoce programas que contengan menos de 12 instrucciones y que satisfagan el planteamiento del problema «c». Sin embargo, se pueden obtener muchas soluciones del problema en cuestión, cuya longitud sea 12. Ante todo, resulta fácil elaborar ¡por lo menos 11! programas, a cuenta de todo género de permutaciones posibles que incluyan todas las instrucciones, salvo la primera, del programa C_1 (sin duda alguna, cada una de estas permutaciones deberá ir acompañada por el cambio de los números y saltos). Además, puede intentarse buscar otros en principio métodos de adición. Por ejemplo, se puede proponer el método que consiste en la múltiple realización del siguiente procedimiento: la registro del número izquierdo se adjunta a la derecha una marca, para cuya compensación se borra una marca en el extremo izquierdo de la escritura del número derecho; este procedimiento continúa hasta que el juego derecho se agote; la marca «sobrante» se vacía al mismo comienzo por la instrucción N° 1. El siguiente programa C_2 de la longitud 12 realiza este método.

Programa C_2

1. ξ 2	5. ⇒ 6	9. ? ↗12 ↘10
2. ⇒ 3	6. ? ↗5 ↘7	10. ⇐ 11
3. ? ↗4 ↘2	7. ξ 8	11. ? ↗10 ↘2
4. ∨ 5	8. ⇒ 9	12. stop.

§ 5. ADICIÓN DE LOS NÚMEROS EN CASOS MÁS COMPLICADOS

Problema «d (k)». Componer para cada k el programa de adición de k números registrados a distancias arbitrarias entre éstos.

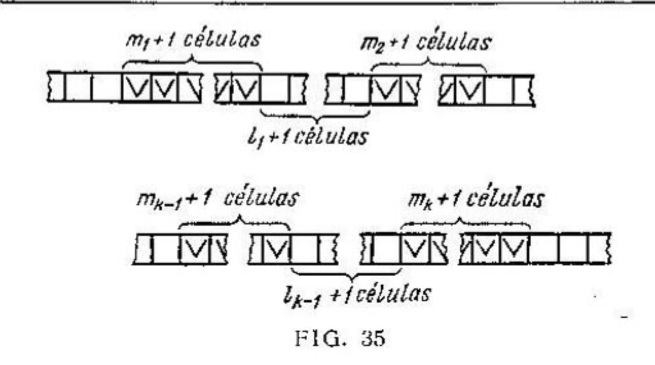


FIG. 35

El estado inicial para el problema «d (k)» se ofrece en la figura 35. Por consiguiente, para cada k es necesario elaborar el programa que lleve al objetivo con cualesquier números no negativos enteros $m_1, m_2, \ldots, m_k, l_1\ l_2, \ldots, l_{k-1}$.

Hallaremos el programa buscado del modo siguiente.

Sea Σ cualquier programa que resulte ser la solución del problema «c» y posea las siguientes propiedades:

1) en el proceso de aplicación del programa Σ al estado inicial del problema «c» (es decir, a tal estado, para el cual en la cinta hay registros de dos números y nada más, mientras que el carro observa la célula más a la izquierda de las que están marcadas) el carro nunca pasará más a la derecha de la célula extrema derecha, entre las marcadas en el estado inicial;

2) una vez terminado el cumplimiento del programa Σ, aplicado al estado inicial del problema «c», resulta que el carro se encuentra frente a la célula más a la izquierda de las que están marcadas en el juego formado;

3) el programa contiene exactamente una sola instrucción de parada, y ésta es la última instrucción del programa.

Sea que el programa Σ contiene $a+1$ instrucciones. Designemos por Ξ la lista de las primeras instrucciones a de Σ. Compongamos tal

programa:

Programa D* (k)

Ξ ⋮

$\Xi\,[+a]$ $\Xi\,[+(k-2)\,a]$

$\Xi\,[+2a]$ $(k-1)\,a+1$. stop.

⋮

Dejemos al lector que compruebe que este programa es la solución del problema «d (k)» (efectivamente, si hacemos $s_i = m_1 + \ldots + m_i$, entonces es fácil de notar que las instrucciones de la lista Ξ realizan la adición de los números m_1 y m_2, las de la lista $\Xi\,[+a]$, la adición de los números s_2 y m_3, etc.; las instrucciones de la lista $\Xi\,[+ia]$, la adición de los números s_{i+1} y m_{i+2}; por último, las instrucciones de la lista

$$\Xi\,[+\,(k-2)\,a]$$

efectúan la adición de los números s_{k-1} y m_k, es decir, la obtención de la suma buscada).

Ejercicio. ¿De qué modo al establecer que el programa D* (k) sirve como solución del problema «d (k)» se utilizan las propiedades 1)...3) del programa Σ?

Nos queda por hallar el programa Σ con las propiedades requeridas. Ninguno de los programas $\tilde{C}$, C_1 y C_2 sirven. Sin embargo, el programa $\tilde{C}$ posee las propiedades 1) y 3). Es fácil de cambiarlo un poco, para que se cumpla también la propiedad 2). Es suficiente en lugar de la última instrucción del programa $\tilde{C}$ escribir las dos siguientes:

14. $\Rightarrow$ 15   15. stop.

En virtud de la observación hecha antes (durante la explicación del funcionamiento del programa $\tilde{C}$) sobre la posición final del carro después de ejecutar el programa $\tilde{C}$, el programa $\tilde{C}^*$ de 15 instrucciones obtenido poseerá no sólo las propiedades 1) y 3), sino que también la propiedad 2).

En calidad de programa Σ como base para elaborar el programa D* (k) se puede tomar además el programa C_2^*, también de la longitud 15, que se obtiene del programa C_2, si en lugar de la última instrucción de este programa escribir estas cuatro:

12. $\Leftarrow$ 13   14. $\Rightarrow$ 15

13. ? $\genfrac{}{}{0pt}{}{14}{12}$ 15. stop.

El programa C_2^* posee las propiedades 2) y 3), mientras que en lugar de la propiedad 1), cierta propiedad más débil (pregunta al lector: ¿cuál?), sin embargo, suficiente para que el programa $D^*(k)$, que se elabora a base del C_2^*, sea la solución del problema «d (k)».

Si en el programa Σ hay 15 instrucciones, en el programa $D^*(k)$ habrá $(k-1)\,14+1=14k-13$ instrucciones.

No obstante, puede estructurarse un programa que corresponda a los requerimientos del problema «d (k)» y contenga una cantidad de instrucciones menor que $14k-13$.

Con este fin, designemos por Δ_i la siguiente lista de instrucciones ($i=0, 1, 2, \ldots$):

$3+9i.$	$\Rightarrow 4+9i$	$8+9i.$	$\Leftarrow 9+9i$
$4+9i.$	$? \begin{smallmatrix} \nearrow 3+9i \\ \searrow 5+9i \end{smallmatrix}$	$9+9i.$	$? \begin{smallmatrix} \nearrow 8+9i \\ \searrow 10+9i \end{smallmatrix}$
$5+9i.$	$\xi\ 6+9i$	$10+9i.$	$\Rightarrow 11+9i$
$6+9i.$	$\Rightarrow 7+9i$	$11+9i.$	$\vee\ 3+9i$
$7+9i.$	$? \begin{smallmatrix} \nearrow 3+9(i+1) \\ \searrow 8+9i \end{smallmatrix}$		

Ahora, designemos por $D(k)$ el siguiente programa:

Programa $D(k)$

1. $\Rightarrow 2$	Δ_0
2. $? \begin{smallmatrix} \nearrow 3 \\ \searrow 1 \end{smallmatrix}$	Δ_1
	$\vdots$
	Δ_{k-2}
	$9k-6$. stop.

Dejemos al lector cerciorarse por sí mismo de que el programa $D(k)$, en primer lugar, es en realidad un programa y, en segundo lugar, el programa que satisface el planteamiento del problema «d (k)». Con ello, en el programa hay $9k-6$ instrucciones. Notemos que tomando $k=2$, obtendremos en forma de $D(2)$ una nueva solución del problema «c», con la particularidad de que esta solución contendrá 12 instrucciones, es decir, tantas, como tienen los programas C_1 y C_2. El programa $D(2)$ trabaja según el mismo método que el programa C_2, consistente en la traslación de las marcas del juego izquierdo al derecho; sin embargo, al cumplir el programa $D(2)$, a distinción de C_2, la marca al principio se borra en el juego derecho y después se añade al izquierdo, con la particularidad de que la última marca del juego derecho se vacía sin que la nueva marca aparezca en el juego izquierdo.

Problema «e». Elaborar el programa de adición de una cantidad tomada al azar de números registrados a distancias arbitrarias entre éstos.

Es evidente, que cada programa, que es la solución del problema «e», servirá al mismo tiempo como solución de cada uno de los problemas anteriores. No obstante, ninguna de las soluciones aducidas antes de los problemas «a», «b», «c» y «d (k)» será la solución del problema «e» (comprueben esto).

El estado inicial para el problema «e» viene expuesto en la figura 35. Sin embargo, ahora es necesario componer el programa **único** que conduzca al objetivo para k **arbitrario.** Intenten hallar semejante programa por su cuenta. Verán que esto no es tan fácil. ¿Pueda ser que el programa requerido no exista en absoluto? En este caso intenten demostrar que éste no existe. La respuesta a la cuestión de si tiene o no el problema «e» solución será dada en el siguiente capítulo.

CAPÍTULO IV. POSIBILIDADES DE LA MÂQUINA DE POST

En el presente capítulo nos ocupará la cuestión de qué cálculos, en general, pueden ser realizados en la máquina de Post. Con este motivo, tendremos que referirnos al concepto general de algoritmo. En el párrafo conclusivo intentaremos comparar la máquina de Post con las máquinas calculadoras electrónicas (ordenadores). La exposición comienza por el análisis de un problema que quedó sin solucionar en el capítulo anterior.

§ 1. ACERCA DEL PROBLEMA DE ADICIÓN DE NÚMEROS A DISTANCIAS ARBITRARIAS

Al final del capítulo anterior fue planteado el problema más general de la adición de los números registrados en la cinta de la máquina de Post, es decir, el problema «e». En este problema se exigía hallar tal programa único para la máquina de Post, que realizara la adición de una cantidad arbitraria de números registrados en la cinta a distancias arbitrarias. Se supuso que salvo estos números, en la cinta nada había escrito y que el carro se encontraba en la célula más a la izquierda de las marcadas. Al final del funcionamiento del programa en la cinta debe ser registrada la suma de todos los números iniciales y nada más (es decir, la cinta restante debe estar vacía).

Recordemos que tenemos comparado con la cinta un «sistema constante de coordenadas», de acuerdo con el cual las células en la cinta están numeradas con números enteros de $-\infty$ a $+\infty$. Para mayor precición consideraremos que todos los números iniciales están escritos en la parte derecha «no negativa» de la cinta, con la particularidad de que la célula más izquierda entre las que están marcadas tiene el número cero. Por lo tanto, en el estado inicial el carro observa la célula nula.

Sin predeterminar por ahora la cuestión de qué aspecto tiene la solución del problema planteado, intentemos analizar esa solución.

Así pues, supongamos que cierto programa es la solución del problema «e» y designémoslo por E. Examinemos el estado de la máquina de Post ofrecido en la fig. 36. En tal estado

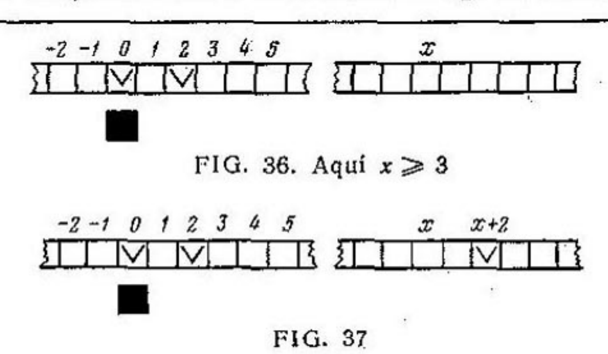


FIG. 36. Aquí $x \geqslant 3$

FIG. 37

en la cinta, distando la unidad uno del otro, están registrados dos números, cada uno de los cuales es igual a 0. Tomemos este estado en calidad de inicial. En este caso, el programa E debe llevar a una parada de resultados, con la particularidad de que en la cinta estará registrado el número 0. Durante el tiempo de trabajo de la máquina, hasta la parada de resultados, el carro ha podido estar sólo en un número finito de células de la cinta. Sea z el mayor entre los números de células, en las cuales ha estado el carro. Designemos por x el mayor entre los números z y 3. Ahora examinemos el estado de la máquina de Post en el que las células 0, 2, $x + 2$ están marcadas, mientras que las restantes, vacías: coloquemos el carro frente a la célula № 0 (fig. 37). Tomemos este estado como inicial y apliquemos a éste el programa E. Veamos cómo éste va a trabajar. Es fácil de ver que éste funcionará «del mismo modo» que en el caso de aplicarlo al estado inicial en la fig. 37.

La expresión «del mismo modo» puesta entre comillas se precisa del modo siguiente. Examinemos dos ejemplares de la máquina de Post. Digamos que cierto estado de la primera máquina es equivalente a cierto estado de la segunda, si en

estos estados 1) el carro de la primera máquina se encuentra frente a la célula con el mismo número que el carro de la segunda; 2) cada célula de la primera máquina, dispuesta a la izquierda de la célula N° $(x + 1)$, está marcada o vacía, simultáneamente con la célula de la segunda máquina que tiene el mismo número; 3) todas las células de la primera máquina, dispuestas a la derecha de la célula N° x, están vacías; 4) la célula N° $(x + 1)$ de la segunda máquina está vacía, la célula N° $(x + 2)$ está marcada, mientras que todas las células dispuestas a la derecha de la célula N° $(x + 2)$ están vacías. Consideraremos que ambas máquinas trabajan según el programa E sincrónicamente, con la particularidad de que en calidad de estado inicial para la primera máquina sirve el estado que se muestra en la fig. 36, en tanto que para la segunda máquina, el estado expuesto en la fig. 37. Estos estados son equivalentes. Mediante la inducción por t se demuestra con facilidad que en cualquier momento de tiempo t será justa la siguiente afirmación: los estados de ambas máquinas en este instante son equivalentes, mientras que la instrucción que debe ejecutar la primera máquina coincide con la que debe cumplir la segunda. Por consiguiente, funcionando de manera sincrónica ambas máquinas pasarán de estados equivalentes a otros equivalentes, bajo la acción de las mismas instrucciones. Precisamente esto se tenía en cuenta, cuando decíamos que el programa aplicado al estado en la fig. 37 trabajaría del mismo modo que siendo aplicado al estado ofrecido en la fig. 36. En cierto instante las máquinas simultáneamente llegarán al estado final en el cual se pondrá en funcionamiento la instrucción de parada. Ambas máquinas realizarán al mismo tiempo paradas de resultados. En la cinta de la primera máquina estará registrado el número 0, es decir, habrá una sola célula marcada, con la particularidad de que esta célula se dispondrá a la izquierda de la célula N° $(x + 1)$. Por consiguiente, en virtud de la equivalencia de los estados finales, en la cinta de la segunda máquina, a la izquierda de la célula N° $(x + 1)$, habrá una célula marcada y, además, tendrá marca la propia célula N° $(x + 1)$, en total, tendremos dos células marcadas.

Hemos llegado a la deducción de que aplicando el programa E al estado expuesto en la fig. 37 surgirá una parada de re-

sultados en el estado, en el cual estarán marcadas tan sólo dos células.

Por otro lado, en el estado ofrecido en la fig. 37 la cinta contiene el registro de tres números, cada uno de los cuales es igual a cero (las distancias entre éstos son iguales a 1 y a x — 1, respectivamente), la cinta restante está vacía y el carro se encuentra frente a la célula más izquierda entre las que están marcadas. Según la suposición el programa E es la solución del problema «e». Por esta razón, al aplicalo al estado inicial, deberá conducir a la parada de resultados en el estado, en el cual en la cinta resultará registrada la suma de números escritos inicialmente, mientras que la cinta restante deberá estar vacía. En el caso dado la suma es igual a 0. Por lo tanto, en el estado final en la cinta estará marcada sólo una célula. Pero, esto se encuentra en contradicción con la deducción obtenida antes.

La contradicción descubierta nos convence de que no puede existir un programa que sea solución del problema «e».

La causa por la cual el problema «e» no tiene solución es muy evidente: por mucho que el carro viaje por la cinta, éste, por así decirlo, «nunca sabrá» si ha recorrido los registros de todos los sumandos o bien, puede ser, que lejos a la derecha se encuentre un sumando más, hasta el cual el carro aún no llegó. Por esta razón, es imposible elaborar el programa que llevara a una parada de resultados con garantía de que todos los sumandos fueron tomados en consideración. Semejante programa puede ser compuesto sólo para casos particulares, por ejemplo, cuando con anticipación es conocido el número de sumandos (como en el problema «d (k)» del párrafo 5 del capítulo anterior) o bien cuando los sumandos están registrados uno respecto del otro a distancias prefijadas (como en el problema «b») o por lo menos restringidas.

Ejercicio. Elaboren para cada q el programa de adición de una cantidad arbitraria de números, registrados uno respecto del otro, a distancias tomadas al azar, pero que no superen q.

En el ejemplo del problema «e» nosotros vemos cuán importante es el procedimiento de registro de los datos iniciales en la cinta.

§ 2. PROPOSICIÓN DE POST

En los capítulos anteriores el lector ha adquirido cierta experiencia de programación en la máquina de Post. Para fijar esta experiencia aconsejamos cumplir los ejercicios 1—5. En éstos se propone al lector elegir por su cuenta la posición inicial del carro y en los ejercicios 2, 3 y 5 también la distancia a la cual se escriben los números iniciales. Por doquier, en el presente capítulo se sobreentiende que los números son enteros no negativos.

Ejercicio 1. Componer el programa de división de los números por 2, 3, 4, ..., por el número prefijado k. **Aclaración.** Por división se entiende la obtención del cociente o cociente incompleto, de manera que el resultado de la división de 7 por 3 será 2.

Ejercicio 2. Elaborar el programa de sustracción de un número del otro. **Aclaración.** Si la resta es imposible (el sustraendo es menor que el minuendo), el programa no deberá llevar a la parada de resultados.

Ejercicio 3. Componer el programa de multiplicación de dos números.

Ejercicio 4. Elaborar el programa de elevación de los números al cuadrado.

Ejercicio 5. Componer el programa de división de un número por otro. **Aclaración.** Si el divisor es 0, el programa no deberá llevar a la parada de resultados.

Para lo futuro nos hará falta un concepto fundamental, aunque simple, a saber, el de cortejo numérico. Con mucha frecuencia se tiene que tratar no con números independenientes, sino que con pares ordenados de números; con ello, se tolera que ambos miembros del par coincidan. Un par ordenado de números lo escribiremos así: al principio, el primer miembro del par, después, el segundo, y todo esto se debe poner entre paréntesis angulares. Así, por ejemplo, $\langle 33, 4\rangle$ y $\langle 2,2\rangle$ son pares ordenados de números. De modo análogo se introduce la noción del terno ordenado de números. Por ejemplo, son posibles semejantes ternos ordenados de números: $\langle 2, 5, 1\rangle$; $\langle 6, 6, 6\rangle$; $\langle 4, 5, 4\rangle$. También $\langle 3, 8, 7, 8, 8, 2, 5\rangle$ es una septena ordenada de números. El juego ordenado de números de

cualquier longitud se denomina *cortejo numérico*; así, los pares ordenados de números, ternos ordenados y la septena ordenada, que acaban de ser apuntados, son los cortejos. El número de «puestos» del cortejo se llama su *longitud*. Así, el cortejo ⟨6, 2, 2, 6⟩ tiene longitud 4.

Todos los problemas y la mayoría de los ejercicios para componer los programas, de los cuales nos ocupábamos en el presente ciclo de artículos, tenían el siguiente aspecto. Se fijaba cierta clase de datos iniciales: números (como en los ejercicios 1 y 4 del presente párrafo o en el problema acerca de la adición de la unidad del capítulo 2), cortejos numéricos de longitud prefijada (longitud dos, como en los ejercicios, 2, 3 y 5 del presente párrafo y en los problemas «a», «c» y «d (2)» del capítulo tercero, o bien, en general, de la longitud k, como en el problema «d (k)» del capítulo indicado), cortejos numéricos de longitud arbitraria (como en los problemas «b» y «e» del capítulo tercero).

A cada uno de los datos iniciales de la clase elegida —número o cortejo — ora se ponía en correspondencia cierto número resultante (el cuadrado del número inicial, o el cociente de la división del primer miembro del cortejo inicial por el segundo, o bien la suma de todos los miembros del cortejo inicial, etc.), ora nada se ponía en correspondencia (por ejemplo, el cociente o el cociente incompleto se ponía en correspondencia sólo a aquel par, en el cual el segundo miembro difiere de cero). Se debía componer un programa, el cual transformara el registro de cualquier dato inicial en la cinta —con la parada de resultados— en el registro del número resultante, si éste existe; pero, siendo aplicado al registro del dato inicial, para el cual no está determinado el número resultante, el programa no deberá dar la parada resultante. Para simplificar siempre sobrentenderemos que en el estado inicial en la cinta está registrado sólo el dato inicial, y el carro se encuentra frente a la célula más izquierda entre las que están marcadas y que en el estado final en la cinta también está escrito el número resultante, mientras que el carro se encuentra dondequiera. Es necesario precisar todavía cómo registrar cortejos numéricos. En el párrafo anterior vimos que del procedimiento de registro del cortejo, y precisamente de las distancias entre sus miembros, depende la misma existencia del programa requerido. Convengamos registrar el cortejo, ubicando las anota-

ciones de sus miembros a distancia de la unidad uno del otro. Al fijar así la posición inicial del carro y el procedimiento de registro de los números y cortejos, se puede hablar en forma abreviada sobre la aplicación del programa al número o al cortejo y su transformación a otro número o cortejo, tomando, con ello, en consideración, en efecto, el registro de los números y cortejos.

Ahora planteemos la siguiente pregunta natural. ¿En qué casos los problemas del aspecto que acabamos de exponer para elaborar los programas tienen soluciones? Con otras palabras: ¿qué cálculos pueden ser realizados en la máquina de Post?

La respuesta a esta pregunta es la siguiente. El problema para componer el programa que lleva del dato inicial al número resultante si, y sólo si, tiene solución, cuando hay algún procedimiento general que permite, a partir del dato inicial arbitrario, registrar el número resultante. Esta afirmación que responde a nuestra pregunta la llamaremos —por el nombre de su autor— *proposición de Post*. Subrayemos que en la proposición de Post se trata de cierto procedimiento único (así mismo como acerca del programa único), que es general para todos los datos iniciales. Además, se supone que si el número resultante no existe, entonces el procedimiento del que se trata (lo mismo que el programa), no lleva a resultado alguno (que es, en este caso, falso a todas luces).

Ejemplo 1. El problema: «Elaborar el programa para la máquina de Post sobre la elevación al cubo» es soluble, porque existe el procedimiento general que permite calcular n^3 por n.

Ejemplo 2. Por esta misma causa existe el programa para la máquina de Post que transforma el registro del cortejo $\langle x, y, z \rangle$ en la inscripción del número $(x^y + zy)(z^2 - y)$. Este programa (como también el respectivo procedimiento de cálculos) no lleva a resultado alguno para $z^2 < y$ (todos los números, con los que tratamos, son enteros no negativos).

Ejemplo 3. Sea que se debe componer un programa para la máquina de Post, que posea la siguiente propiedad: si en el desarrollo decimal del número π se encuentran seguidamente $n + 2$ signos decimales, entre los cuales sólo el primero y el último difieren del nueve, mientras que los n restantes son iguales al nueve, entonces el programa transforma este número n en 1; mas si la sucesión finita expuesta de $n + 2$ signos en

el desarrollo del número π no se encuentra, el programa transforma n en 0. En el caso dado, aunque a cada n está puesto en correspondencia cierto número resultante ξ_n que es igual a 0 ó 1, para nosotros no es conocido el procedimiento general, que permita calcular este número ξ_n para n arbitrario. El análisis de los primeros 800 signos de desarrollo del número π[1] muestra sólo que $\xi_n = 1$, para $n = 0, 1, 2, 6$. En general, por muchos que sean los signos decimales de desarrollo del número π que anotemos, de la sucesión obtenida de signos podemos sacar sólo la información sobre la igualdad del número ξ_n a la unidad para ciertos valores de n y ninguna información acerca de la igualdad del ξ_n a cero, aunque sea para algún n. La igualdad de $\xi_n = 0$ para cierto valor de n, incluso si puede establecerse, será sólo mediante un método indirecto. El procedimiento general de cálculo del número ξ_n con cualquier n no es conocido para nosotros. Si tal procedimiento no existe, entonces en virtud de la proposición de Post tampoco puede existir el programa requerido. (Notemos que el empleo de la proposición de Post hacia este lado es evidente: si existiera el programa exigido, éste mismo daría cierto procedimiento general de cálculo del número resultante por cualquier n.) Pero, si tal procedimiento existe, entonces en virtud de la proposición de Post (cuya aplicación hacia este lado ya no es evidente) también existe el programa requerido.

Ejemplo 4. Determinemos para cada número n el número ζ_n del modo siguiente: $\zeta_n = 1$, si para todo k en el desarrollo decimal del número π se encuentran seguidamente k nueves (al igual que se encuentran 0, 1, 2, 3, 4, 5, 6 de semejantes nueves); $\zeta_n = 0$ en caso contrario, es decir, si existe la máxima longitud del juego de los nueves que siguen uno tras otro (si esta máxima longitud existe, entonces, sin duda alguna, será más de 5). Se pregunta existe o no el programa para la máquina de Post que transforma cada n en ζ_n. La respuesta a esta pregunta puede parecer un poco inesperada: «Sí, existe, aunque no podemos indicarla». Efectivamente, ora en el desarrollo del número π se puede encontrar cualquier número de nueves que siguen uno tras otro, ora esto no es así. En cada

[1]) Estos signos se aducen en la pág. 63 del libro de *Littsman V.*, Gigantes y enanos en el mundo de números, Editorial «Fizmatguiz», Moscú, 1959, (en ruso).

uno de estos casos existe el procedimiento general de cálculo del número ζ_n: hay que tomar $\zeta_n = 1$ (para todos n) en el primer caso y $\zeta_n = 0$ (para todos n) en el segundo. Otra cosa es que no sabemos, cuál de estos dos procedimientos generales puede ser elegido, puesto que nos es desconocido, cuál de los dos casos respecto al desarrollo del número π tiene lugar en realidad. Para cada uno de estos procedimientos generales de cálculo del número ζ_n existe el respectivo programa que efectúa el cálculo del número ζ_n en la máquina de Post, es decir, para el primer procedimiento general ($\zeta_n \equiv 1$), el programa:

1. ξ 2
2. $\Rightarrow$ 3
3. ? $\begin{smallmatrix} \nearrow 4 \\ \searrow 1 \end{smallmatrix}$
4. $\Leftarrow$ 5
5. $\vee$ 6
6. $\Leftarrow$ 7
7. $\vee$ 8
8. stop,

mientras que para el segundo procedimiento general ($\zeta_n \equiv \equiv 0$), el programa:

1. $\Rightarrow$ 2
2. ? $\begin{smallmatrix} \nearrow 4 \\ \searrow 3 \end{smallmatrix}$
3. ξ 1
4. stop.

Podemos afirmar con seguridad que uno de los dos programas escritos corresponde a los requerimientos planteados, pero el estado actual de nuestros conocimientos no nos permite decir, cuál de ellos precisamente.

La existencia del procedimiento único correspondiente de cálculo fue conocida en los ejemplos 1, 2 y 4 y quedó desconocida en el ejemplo 3. Se han compuesto ejemplos, en los cuales este procedimiento no puede existir a ciencia cierta, pero éstos son muy complicados para que puedan ser expuestos aquí.

§ 3. MÁQUINA DE POST Y ALGORITMOS

La noción del procedimiento único de cálculo, que es general para una clase entera de posibles datos iniciales, representa de por sí uno de los conceptos más importantes de las

matemáticas tanto «teóricas», como «cotidianas». Precisamente en este procedimieto se basa la enseñanza de las matemáticas en la escuela primaria. Varios ejemplos de las cuatro operaciones con números polidígitos no persiguen, en verdad, el objetivo de enseñar a sumar, restar, multiplicar y dividir precisamente aquellos números que se encuentran en estos ejemplos. La finalidad de estos ejercicios es practicar los métodos de adición, sustracción, multiplicación y división en columna, teniendo lógicamente en cuenta, que estos métodos son aplicables a *cualesquier* pares ordenados de números y no sólo a los que hay en el compendio de problemas y ejercicios. Cuando dicen que un escolar sabe sumar, sobrentienden no sólo que él hallará tarde o temprano para cualquier par de números su suma, sino que él domina el procedimiento general de adición.

Es lógico, que un concepto tan importante requiere un término especial para su denominación: en calidad de tal término se utiliza la palabra «algoritmo» (otro modo de ortografiarla: «algorifmo»). Ahora podemos decir que en los ejemplos 1, 2 y 4 existen algoritmos (aunque o sabemos cuál es éste para el ejemplo 4), en el ejemplo 3 el algoritmo no es conocido (incluso no se sabe si existe éste o no). Como ya se indicó, son conocidos casos, cuando a ciencia cierta no existe el algoritmo requerido.

Aunque con la noción de algoritmo se tropieza a menudo en la práctica y en las ciencias ésta no siempre se percibe: la propia palabra «algoritmo» no es aún suficientemente popular; es de común saber sólo la combinación de palabras el «algoritmo de Euclides», que sirve para designar uno de los algoritmos a fin de hallar el máximo común divisor. Por eso, a menudo, nos encontramos en la situación del personaje de la obra de Molière «El burgués gentilhombre», quien ya en edad madura llegó a saber que toda su vida hablaba en prosa. Como quiera que sea, ahora sabemos que en la escuela nos enseñaron precisamente algoritmos.

Se supone que para cada algoritmo está indicado cierto conjunto que se llama *dominio* de sus *posibles datos iniciales* y que consta de todos los objetos (por ejemplo números o cortejos), a los cuales tiene sentido intentar la aplicación del algoritmo que se examina. Al ser aplicado a cualesquiera de sus posibles datos iniciales el algoritmo puede tanto dar re-

sultado (en este caso se dice que el algoritmo es *aplicable* a este dato inicial y lo transforma en resultado), como no darlo (en este caso se dice que el algoritmo es *inaplicable* a este dato inicial). Así, para el algoritmo de sustracción en columna sirven, como datos iniciales, pares ordenados de números, mientras que el algoritmo es aplicable sólo a aquellos de ellos cuyo segundo término no es mayor que el primero. Es poco probable que sea conveniente considerar que para este algoritmo la clase de posibles datos iniciales consta de los pares (a, b) solamente, en los cuales $a \geqslant b$. En verdad, es posible intentar efectuar la sustracción del segundo número a partir del primero también para el par

$$(1244444445444445, 1244444454444445);$$

simplemente, en este caso, no obtendremos resultado alguno.

Veamos, cómo el concepto de algoritmo está ligado con la máquina de Post. Examinemos tales programas para la máquina de Post que, siendo aplicados a cualesquier número, ora no dan en absoluto la parada de resultados, ora dan de nuevo como resultado un número (es decir, si ocurre una parada de resultados, entonces en la cinta resulta registrado cierto número y sólo éste). Cada semejante programa prefija el siguiente algoritmo, para el cual sirve de dominio de posibles datos iniciales el conjunto de todos los números no negativos enteros y todos los resultados también son números: es necesario tomar el número inicial, registrarlo en la cinta de la máquina de Post, instalar el carro frente a la célula más izquierda entre las que están marcadas, poner en marcha la máquina según el programa examinado, esperar la parada de resultados y leer el número que resultará escrito en la cinta después de esta parada.

Ahora fijemos cualquier número k y examinemos aquellos programas que siendo aplicados a cualesquier cortejo numérico de la longitud k ora no dan, en absoluto la parada de resultados, ora dan como resultado un número. Cada semejante programa prefija el siguiente algoritmo, para el que sirve de dominio de posibles datos iniciales el conjunto de todos los cortejos numéricos de la longitud k, mientras que todos los resultados son números: hay que tomar el cortejo numérico inicial, registrarlo en la cinta de la máquina de Post, colocar el carro frente a la célula más izquierda entre las que están

marcadas, poner en marcha la máquina de acuerdo con el programa examinado, esperar la parada de resultados y leer el número que estará registrado en la cinta después de esta parada.

De modo análogo, cada uno de los programas para la máquina de Post, que siendo aplicado a todo cortejo numérico de **longitud cualquiera** ora no da la parada de resultados, ora da como resultado un número, prefija cierto algoritmo.

Designemos la serie natural por la letra N, el conjunto de todos los cortejos numéricos, por el símbolo N^{∞} , el conjunto de todos los cortejos numéricos de la longitud k, por el símbolo N^{k}.

Ahora con ayuda del término «algoritmo» la proposición de Post puede ser enunciada así.

Sea dada cierta clase de datos iniciales —N, N^{k} o bien N^{∞} . Sea que a cada dato inicial de esta clase ora nada está puesto en correspondencia, ora está puesto en correspondencia cierto número resultante (hablando en general, el inherente a cada dato inicial). Examinemos dos problemas: (P) «Componer un programa para la máquina de Post que va a transformar cualquier dato inicial, para el cual hay número resultante, en este número y no llevará a ningún resultado para el dato inicial, para el cual no hay número resultante»; (A), el programa que se obtiene de (P) al sustituir las palabras «programa para la máquina de Post» por la palabra «algoritmo». Entonces los problemas (P) y (A) simultáneamente ora no tienen solución, ora la tienen.

Hasta ahora no basamos la proposición de Post de manera alguna, sino que la aceptamos a pie juntillas. Ahora veremos que esto, es, en cierto grado, inevitable. En efecto, la proposición de Post consta, en realidad, de dos afirmaciones: en la primera se dice que de la decidibilidad del problema (P) se desprende la del problema (A), en la segunda se dice que de la solubilidad del problema (A) se desprende la del problema (P). La primera de estas afirmaciones es evidente, en vista de que el programa que sirve como solución del problema (P) por sí mismo forma el algoritmo que sirve como solución del problema (A); sobre esto ya se habló durante el examen del ejemplo 3. En lo que se refiere a la segunda afirmación, la cual sólo necesita argumentación (y a la cual se reduce, en realidad, la proposición de Post), ésta fue enunciada, en expresiones al-

go diferentes, por Emil Post en su célebre artículo «Procesos combinatorios finitos, enunciación 1» en calidad de «hipótesis de trabajo».

Esta «hipótesis de trabajo», la que denominaremos también hipótesis de Post, no puede ser demostrada, por lo menos en aquel sentido de la palabra «demostración», que está adoptada en matemáticas y ni mucho menos porque aquélla sea incorrecta, sino porque el concepto general del algoritmo, que toma parte en esta hipótesis no está determinado «matemáticamente». Recordemos que en el presente párrafo ya hemos determinado el algoritmo como «único procedimiento de cálculo, que es general para toda la clase de posibles datos iniciales». También son conocidas otras definiciones análogas: algoritmo o bien algorifmo es «cualquier sistema de cálculos, que se efectúan por reglas determinadas rigurosamente, el cual después de cierto número de pasos lleva, a ciencia cierta, a la solución del problema planteado»[1]); es el «proceso de cálculo, que se realiza según la orden exacta y que lleva de los datos iniciales que pueden variar al resultado buscado»[2]); es la orden exacta, que determina el proceso de cálculo y lleva de los datos iniciales variables al resultado buscado[3]).

Claro está que todas estas definiciones no son matemáticas, sino, más bien, las **descripciones** de la noción de algoritmo. (En lo que se refiere a la ausencia de una determinación completa «rigurosa», el concepto de algoritmo se parece al de demostración; por cierto, el concepto de demostración no tiene

---

[1]) *Kolmogórov A. N.*, Algoritmo, Enciclopedia Soviética Grande, 2ª edición, vol, 2, p. 65, Editorial «Enciclopedia Soviética», Moscú, 1950, (en ruso). (*Uspenski V. A.*, Algoritmo, «Enciclopedia matemática», vol. 1, p.p. 202...206, Editorial «Enciclopedia Soviética», Moscú, 1977, (en ruso); una definición más del concepto de algoritmo se ofrece en el «Diccionario Enciclopédico Salvat Universal», Barcelona, Salvat Editores, 1969, vol. 2, p. 14: «Procedimiento general por medio del cual se puede encontrar la solución o respuesta a cada problema o pregunta de una clase, de un modo puramente mecánico y en un número finito de pasos»).

[2]) *Márkov A. A.*, Teoría de los algorifmos, Obras del Instituto de matemáticas V. A. Steklov de la AC de la URSS, 38, p. 176, Editorial «Academia de Ciencias de la URSS», Moscú, 1951, (en ruso).

[3]) *Márkov A. A.* Teoría de los algorifmos, Obras del Instituto de matemáticas V. A. Steklov de la AC de la URSS 42,p.3. Editorial «Academia de Ciencias de la URSS» Moscú, 1954, (en ruso).

no sólo definición, sino incluso descripciones satisfactorias[1].)

En la fase actual de desarrollo de las matemáticas es imposible «demostrar con rigor» la hipótesis de Post, basándose en semejante género de descripciones del algoritmo.

La argumentación de la «hipótesis de trabajo» de Post se efectúa por otra vía, más habitual para el naturalista, que para el matemático, es decir, por la vía del experimento reforzado con razonamientos especulativos. El experimento muestra que efectivamente cada vez, cuando nos indican el algoritmo, éste puede ser convertido en forma de un programa para la máquina de Post, que puede llevar al mismo resultado. Los razonamientos especulativos, no los aduciremos aquí, confirman que las ideas que tienen ahora los matemáticos acerca de la noción del procedimiento de cálculo, es decir, del algoritmo, no prestan la posibilidad de elaborar un algoritmo, que no pudiera ser sustituido por el programa de la máquina de Post. El mismo Post veía el éxito de sus ideas en la transformación de su «hipótesis de trabajo» en una «ley de la naturaleza». Se puede considerar que esta transformación ya ocurrió.

Para concluir este párrafo aduciremos una enunciación más de la hipótesis de Post. Examinemos cualesquiera dos algoritmos, en los cuales coincide el conjunto de sus posibles datos iniciales. Denominemos a estos dos algoritmos *equivalentes*, si para todo dato inicial, del conjunto común para ellos, los dos resultan ser aplicables, o bien ambos inaplicables y en caso de que sean aplicables, dan un mismo resultado. El

---

[1]) Hoy día, a mediados del siglo XX, como en la época de Euclides, la demostración matemática sigue siendo no más de un razonamiento convincente que puede cerciorarnos hasta tal punto, que nosotros estemos dispuestos a convencer con su ayuda a otros. La formalización del concepto de demostración con los medios de la lógica matemática, con toda su importancia, no permite liquidar completamente el empleo de esta noción en su entendimiento intuitivo recién expuesto. El gran descubrimiento hecho en los años 30, por el eminente matemático y lógico de nuestro tiempo Kurt Gödel, consiste en que la tentativa de precisar formalmente el concepto de demostración resulta de modo inevitable incompleto: se descubren verdades que se demuestran por intuición, pero no permiten efectuar demostraciones entre los límites de la precisión realizada. (Vease *Uspenski V. A.*, Teorema de Gödel sobre la incompletitud en una exposición elemental, «Exitos de las ciencias matemáticas», 1974, 29, Nº 1, p. p. 3...47, en ruso.)

concepto de equivalencia permite enunciar la hipótesis de Post del modo siguiente: cada algoritmo, todos los resultados del cual son números, y como dominio de los posibles datos iniciales sirven N, N^k o bien N^∞ , es equivalente al algoritmo con el mismo conjunto de posibles datos iniciales, prefijado por cierto programa para la máquina de Post (que se prefija así, como fue explicado en las páginas 75 y 76).

§ 4. CONCEPTO SOBRE LA FUNCIÓN COMPUTABLE

Este párrafo está dedicado a una de las nociones centrales de la teoría de los algoritmos, es decir, al concepto de función calculable. Éste plantea requerimientos un poco más altos a la preparación del lector, precisamente, se supone la familiarización de este último con las nociones de conjunto y de correspondencia.

Ante todo, notemos que la función computable es la que se calcula[1]. Ahora hay que explicar las palabras «función» y «computable».

Llámase función del conjunto A en el conjunto B la correspondencia entre los elementos A y B, con la cual a cada elemento A corresponde no más de un elemento B (pero también puede no corresponder ni un solo). Con ello, el conjunto A se denomina dominio de partida, mientras que el conjunto A, dominio de llegada de la función examinada. A veces se habla de una función como de una «regla» o «ley», que permite comparar con ciertos elementos A los correspondientes elementos B, pero, quizá estos términos son menos acertados, ya que pueden crear una imagen incorrecta sobre que la regla (=ley) deben (o pueden) ser escritas de alguna manera. Pero, en realidad, esto no es así: cualquier que sea el procedimiento de descripción de las funciones, por ejemplo, con ayuda de las oraciones en el idioma español complementado, si es necesario, con símbolos matemáticos que nosotros eligiésemos, siempre habrán funciones que no tienen descripción[2].

En calidad del dominio de partida examinaremos uno de los conjuntos N, N^k, y N^∞ [por lo demás, podemos identificar los números naturales con los cortejos (sucesiones finitas) de la longitud 1 y, de esta manera, no distinguir los conjuntos N y N^1], mientras que en calidad del dominio de llegada, tomar el conjunto N. Así pues, servirán como argumentos de las funciones, que nosotros estamos examinando, los números naturales (es decir enteros no negativos) y los cortejos numéricos, en tanto que como valores, los números naturales. En

[1]) Esta nota no es tan trivial como parece: por ejemplo, muy señor mío—de ningún modo quiere decir que ese señor es mío.

[2]) Este hecho puede ser demostrado aplicando el mismo método, mediante el cual, a continuación, será establecida la existencia de la función no calculable.

calidad de función de género semejante puede ser aducida la función «sustracción». Esta función de N^2 en N confronta con el cortejo $\langle x, y\rangle$ el número $x-y$, si $x \geqslant y$, y no confronta nada, si $x < y$. Por regla, las funciones de N^k en N se llaman numéricas. Ahora tenemos que explicar qué funciones consideraremos computables. Durante esta explicación deberemos utilizar el concepto de algoritmo. Por esta razón, la noción de la función calculable resultará no más (pero, también no menos) «matemáticamente» determinada que el concepto de algoritmo.

Pues bien, sea que hay un algoritmo A, para el cual en calidad del campo de los datos iniciales posibles sirve ora el conjunto N^k, para cierto $k \geqslant 1$, ora el conjunto N^∞. Sea que el resultado de aplicación del algoritmo A a todo dato inicial posible (si es que tal resultado existe) siempre es un número natural. En este caso, podemos determinar la función f de N^k en N (si como campo de los datos iniciales posibles servía N^k) o bien de N^∞ en N (si como campo de los datos iniciales posibles servía N^∞) así: a los cortejos $\langle x_1, \ldots, x_k\rangle$, para los cuales el resultado de aplicación del algoritmo A (a ellos) existe, la función pone en correspondencia este resultado, mientras que a los cortejos restantes la función nada pone en correspondencia. En otras palabras,

$$f(x_1, \ldots, x_k) = \begin{cases} \text{resultado de aplicación de } A \text{ a } \langle x_1, \ldots, x_k\rangle, \\ \qquad \text{si } A \text{ se aplica a } \langle x_1, \ldots, x_k\rangle \\ \text{no se determina, si } A \text{ no se aplica a} \\ \qquad \langle x_1, \ldots, x_k\rangle. \end{cases}$$

Sobre la función f construida así se dice: «el algoritmo A calcula la función f". Es evidente, que cada algoritmo calcula sólo una función. Sin duda alguna, distintos algoritmos pueden calcular una misma función.

Ahora podemos enunciar lo siguiente

Definición de la función computable. La función f de N^k en N° de N^∞ en N se denomina calculable, si existe el algoritmo que la calcula. Dicho de otra manera: una función f es calculable si, y sólo si, hay un algoritmo que sirve para calcular el valor correspondiente a cada uno de sus argumentos.

Según esta definición la función «sustracción» examinada antes es calculable—el correspondiente algoritmo («sustracción en columna») se estudia en la escuela primaria.

La definición enunciada del concepto de la función calbulable es, más bien, de ciencias naturales, que matemáticas. Si consideramos que esto es una insuficiencia y deseamos eliminarla, debemos sustituir la noción común (e indeterminada en algo) de algoritmo por otra más exacta (pero, en consecuencia, más estrecha) de «algoritmo de género especial». En calidad de semejantes «algoritmos de género especial» se pueden tomar los programas para la máquina de Post. En este caso, obtendremos lo siguiente:

Definición de la función computable en la máquina de Post. La función f de N^k en N o de N^∞ en N se llama computable en la máquina de Post (o bien calculable según Post), si existe el programa p para la máquina de Post que calcule la función f, es decir, el programa que posea estas propiedades:

(1) si $f(x_1, \ldots, x_k) = y$, entonces el programa p, que se aplica al dato inicial $\langle x_1, \ldots, x_k \rangle$, termina el trabajo, después de lo cual en la cinta queda el registro del número y;

(2) si $f(x_1, \ldots, x_k)$ no está determinada, entonces la aplicación del programa p al dato inicial no lleva a la parada de resultados.

Ahora la proposición de Post puede ser enunciada así:

Sea dada cierta clase de los datos iniciales: N, N^k $(k > 1)$ o bien N^∞; designemos esta clase por A. Sea prefijada la función f de A en N. Examinemos dos problemas: (P) «Elaborar un programa para la máquina de Post, que calcule la función f»; (A) «Componer el algoritmo, que compute la función f». Entonces los problemas (P) y (A) simultáneamente ora tienen solución, ora no la tienen.

Al hacer uso del concepto de función computable y la noción de función calculable en la máquina de Post, esta misma afirmación puede ser reenunciada así:

Tesis de Post. Sea A uno de los conjuntos N, N^k y N^∞. En este caso la clase de las funciones calculables de A en N coincide con la clase de las funciones de A en N computables en la máquina de Post.

Expliquemos, por qué a esta enunciación de la hipótesis de Post la hemos llamado «tesis». Esto se explica por las siguientes causas históricas.

Tropezamos aquí con la siguiente situación. Hay una definición matemática exacta de cierta clase de funciones numéricas (en el caso dado, de la clase de funciones calculables en la máquina de Post). Después de esto se afirma que cualquiera función computable (en el sentido de la definición «de ciencias naturales») pertenece a esta clase. Por primera vez una afirmación de semejante género fue hecha por Alonzo Church en 1936 y, a continuación, obtuvo el nombre «tesis de Church». Esta tesis afirmaba que toda función numérica calculable dondequiera determinada es en general recursiva, con la particularidad de que la clase de las funciones en general recursivas fue descrita por medio de una definición matemática estricta. En lo sucesivo, Stephen Kleene generalizó esta afirmación, planteando otra tesis («tesis de Church—Kleene»): cualquiera función computable (no obligatoriamente determinada dondequiera) es recursiva parcialmente[1]). La tesis de Post se distingue de la de Church—Kleene, en que en lugar de las funciones recursivas parcialmente en la primera se examinan las funciones calculables en las máquinas de Post. Aunque ambas estas afirmaciones tienen carácter «de ciencias naturales» su equivalencia puede ser demostrada matemáticamente: precisamente se puede demostrar que cualquiera función recursiva parcialmente es calculable en la máquina de Post y, al contrario, cualquiera función que se computa en la máquina de Post es recursiva parcialmente.

Por lo tanto, la tesis de Post resultó equivalente a la de Church — Kleene. En este hecho se puede discernir un argumento adicional en

favor de su justeza. Son conocidas también otras afirmaciones análogas a la tesis de Post. Ellas surgen, cuando en calidad de «algoritmos de género especial» se toman no programas para las máquinas de Post, sino que algoritmos de otra clase representada claramente («máquina de Turing», «algoritmos normales de Márkov», «algoritmos de Kolmogórov», etc.). Todas estas afirmaciones resultan equivalentes entre sí.

Ahora nos ocuparemos del estudio de la siguiente cuestión: ¿son calculables o no todas las funciones numéricas? Si lo son todas, entonces la teoría de las funciones computables pierde su contenido. Afortunadamente (por lo menos para esta teoría), esto no es así, como ya se indicó, existen funciones que no son calculables, y pronto lo veremos.

Pero, primero, para familiarizarnos con el concepto de la función computable estudiaremos una vez más los ejemplos del § 2 del presente capítulo.

En el ejemplo 1 se explica que la función f de N en N, para la cual $f(n) = n^2$, es calculable.

En el ejemplo 2 se examina la función g de N^3 en N, para la cual

$$g(x, y, z) = \begin{cases} (x^y + zy)(z^2 - y), & \text{si } z^2 \geqslant y \\ \text{no está determinada}, & \text{si } z^2 < y \end{cases}$$

y se destaca su calculabilidad.

En el ejemplo 3 se examina la función

$$h(n) = \begin{cases} 1, & \text{si en el desarrollo decimal del número } \pi \text{ hay un segmento de } n \text{ nueves rodeado no de nueves} \\ 0, & \text{en el caso contrario} \end{cases}$$

y se dice, que no se sabe si esta función es calculable o no.

---

[1]) Llámanse recursivas parcialmente las funciones numéricas que pueden ser obtenidas de la función cero idéntica y la función de adición de la unidad ($f(x) = x + 1$) con ayuda de las operaciones 1) sustitución (de una función en otra), 2) determinación recurrente (=inductiva = recursiva) de la función y 3) representación implícita de la función. Hablando en general, las funciones recursivas parcialmente no son determinadas dondequiera, en vista de que la representación implícita [es decir, el paso de la función $f(x_1, \ldots, x_k, y)$ a la función $\varphi(x_1, \ldots, x_k)$ es tal que $f(x_1, \ldots, x_k, \varphi(x_1, \ldots, x_k)) = 0$] puede llevar a la función φ), que no es determinada dondequiera. Las funciones en general recursivas pueden ser caracterizadas como aquellas funciones recursivas parcialmente, que son determinadas dondequiera. Más detalladamente véase en el artículo «Funciones recursivas» en la Enciclopedia Soviética Grande (en ruso; 2ª edición, vol. 36; 3ª edición, vol. 21).

En el ejemplo 4 se examina la función

$$p(n) = \begin{cases} 1, & \text{si para cualquier } k \text{ en el desarrollo de } \pi \text{ se encuentra} \\ & \text{el segmento de } k \text{ nueves;} \\ 0, & \text{si esto no es así} \end{cases}$$

y se demuestra que esta función es computable[1]).

Así pues, las funciones de los ejemplos 1, 2 y 4 son calculables. Si nosotros creemos en la realidad de la hipótesis de Post, entonces también debemos creer que estas funciones son computables en la máquina de Post. Y esto es efectivamente así, es decir, pueden ser escritos los programas para la máquina de Post, que calculan estas funciones. Por ejemplo, para la función del ejemplo 4 el programa buscado es uno de los dos programas aducidos en el fin del § 2 (aunque, cual de los dos, precisamente, no se sabe).

Ahora, habiéndonos acostumbrado al concepto de función computable, deseamos componer un ejemplo de una función numérica, no calculable. Con mayor precisión, deseamos confeccionar un ejemplo de la función numérica que no se calcula en la máquina de Post. Para ello, necesitaremos tal afirmación:

Existe la sucesión p_0, p_1, . . . de programas para la máquina de Post, que contiene todos los posibles programas para la máquina de Post.

(La reenunciación para los conocedores: el conjunto de los programas para la máquina de Post es numerable.)

La demostración de esta afirmación es muy fácil. En efecto, la cantidad de programas para la máquina de Post de la longitud fijada dada es finita (véase el ejercicio 1 del § 2 del capítulo primero). Al apuntar por orden primeramente todos los programas de la longitud 1 (en cualquier orden), luego, todos los programas de la longitud 2 (en cualquier orden), etc., obtendremos la sucesión buscada.

---

[1]) Si la demostración ofrecida allí no eliminó por completo sus dudas de que esta función es calculable, reflexione en cuál, precisamente, de las cuatro afirmaciones siguientes Ud. duda.

A. La función v determinada así: $v(n) = 0$, para todos los n es calculable.

B. La función w determinada así: $w(n) = 1$, para todos los n es computable.

C. La función p ora es igual a v, ora es igual a w ($p = w$, si en el desarrollo π hay segmentos cuanto quiera largos de los nueves y $p = v$, en el caso contrario).

D. Por lo tanto, p es calculable.

Si duda en la afirmación C, entonces Ud. es el correlegionario excelente del matemático holandés L. E. J. Brouwer y de la escuela de los intuicionistas fundada por él. (Acerca del intuicionismo véase, por ejemplo, *Heyting A.* Intuitionism. Amsterdam, 1956. Así como también: *Dragalin A. G.*, Intuicionismo matemático. Introducción a la teoría de las demostraciones, Moscú, Editorial «Naúka», 1979, 1ª parte, p. 1 «Explicaciones no formales», en ruso).

Ahora procedamos a componer la función f de N en N, que no se calcula en la máquina de Post. En otras palabras, necesitamos confeccionar tal función f, que ningún programa para la máquina de Post la calcule. Obremos así: sea que f traslada el número i adondequiera, sólo que no al resultado de aplicación del programa p_i al número i.

Con la mayor precisión determinemos f, por ejemplo, así:

$$f(n) = \begin{cases} 0, & \text{si la aplicación del programa } p_n \text{ a } n \text{ no lleva a la parada de resultados o bien, si el resultado no es el registro de un número natural;} \\ k+1, & \text{si la aplicación del programa } p_n \text{ al número } n \text{ da un resultado que es el registro del número } k. \end{cases}$$

Demostremos que la función confeccionada de semejante manera no se calcula en la máquina de Post. En efecto, sea p cualquier programa para la máquina de Post. El programa p se encuentra entre los programas $p_0, p_1, \ldots$: $p = p_s$ con cierto s. ¿Qué se obtiene al aplicar el programa p_s y la función f al número s? Son posibles dos casos.

Primer caso. La aplicación del programa p_s al registro del número s lleva a la parada de resultados y el resultado es el registro del número natural k. Entonces $f(s) = k + 1$ y, puesto que $k \neq k + 1$, el programa p_s no calcula f.

Caso segundo. La aplicación del programa p_s al registro del número s no lleva a la parada de resultados o bien el resultado no es el registro de algún número natural. Entonces p_s no computa f por lo menos porque la función f es determinada dondequiera.

Así pues, en ambos casos se esclarece que el programa p $(= p_s)$ no computa la función f. Lo que se quería demostrar.

Del razonamiento recién aducido empieza la teoría de las funciones calculables. Esta teoría puede desarrollarse de modo matemático puro, sin hablar nada del concepto de ciencias naturales sobre el algoritmo y sin citar la tesis de Post, como la teoría de las funciones que se calculan en la máquina de Post.

¿Para qué sirve entonces la tesis de Post? He aquí para qué. En primer lugar, esta tesis atribuye a la teoría matemática de las funciones computables en la máquina de Post un sentido filosófico, transformándola en la teoría *general* de las funciones calculables. En segundo lugar, esta tesis juega un papel heurístico, permitiendo adivinar la calculabilidad de la función en la máquina de Post antes de elaborar el programa que la computará.

§ 5. MÁQUINA DE POST Y ORDENADORES ELECTRÓNICOS

¿En qué consiste la semejanza y la distinción entre la máquina de Post y la máquina calculadora electrónica (ordenador)?

Ante todo ocupémonos de la semejanza que tiene lugar, por lo menos, en los tres siguientes aspectos:

1. En el ordenador, como en la máquina de Post, se pueden destacar portadores de información (unidades de memoria) «atómicos» (es decir, los que posteriormente son indivisibles; en la máquina de Post en calidad de semejantes portadores atómicos sirven las células de la cinta): como también en la máquina de Post, cada uno de tales portadores de información puede encontrarse en uno de los dos estados (en la máquina de Post, «vacío» o bien «marcado»); toda la información almacenada en la máquina en el momento dado aparece en forma de distribución de estos estados entre los portadores elementales.

2. Para el ordenador, como también para la máquina de Post, se indica cierto juego limitado de operaciones elementales; por un paso el ordenador, como la máquina de Post, puede realizar cualquiera operación de este juego.

3. El ordenador, lo mismo que la máquina de Post, funciona a base de una instrucción especial, es decir, el programa que indica, cuáles operaciones elementales y en qué orden deben ser efectuadas.

Por consiguiente, para registrar en la máquina una información hay que identificarla con cierta combinación de estados de los portadores elementales. Y para realizar el algoritmo en la máquina, es necesario representarlo en forma de la sucesión de ciertas operaciones elementales. Precisamente en esto consiste el arte del creador de los programas, es decir, del programista. Mientras que el arte del diseñador de la máquina consiste, en particular, en el pertrechamiento de ésta con tal juego de operaciones elementales, que los algoritmos, destinados a ser procesados en esta máquina, se descompongan cómodamente en estas operaciones.

Al mismo tiempo queremos llamar la atención del lector sobre los siguientes rasgos esenciales del ordenador, que en la máquina de Post no existen en absoluto, o bien en ella no se revelan debidamente.

1. La información en la máquina de Post, almacenada en el «dispositivo de memoria», o simplemente «memorizador», (es decir, en el dispositivo destinado para el almacenamiento de la información; en el caso de la máquina de Post semejante dispositivo es la cinta dispuesta linealmente. Esto significa que cada célula de la cinta tiene sólo dos «vecinos», al procesamiento de los cuales la máquina puede pasar después de

procesar la célula dada. En el ordenador uno u otro «sector» del dispositivo de memoria puede tener, hablando en general, muchos más sectores «vecinos», es decir, lindantes en el sentido que la máquina puede pasar al procesamiento de cualquiera de ellos una vez procesado el sector inicial. Las ventajas de tal organización del dispositivo de memoria son evidentes, por lo menos desde el punto de vista de reducir el número de pasos del trabajo del ordenador, ya que en la máquina de Post el carro para pasar de la observación de una célula a la observación de otra tiene que atravesar, obligatoriamente, todas las células intermedias. Si hablamos con más detalles es preciso decir que el dispositivo de memoria de un ordenador, por regla, consta de dos unidades: de memoria «externa» de gran capacidad que almacena la información linealmente de manera parecida a la cinta de la máquina de Post y de memoria «interna» de una capacidad relativamente pequeña, pero con más ricas «comunicaciones de vecindad» entre los sectores independientes de esta unidad.

2. En el caso de la máquina de Post nada dijimos de cómo, precisamente, se asegura el cumplimiento del programa. Se puede pensar, por ejemplo, que junto a la máquina de Post se encuentra una persona que lee el programa registrado en el papel y siguiéndolo traslada el carro, imprime y borra la marca. Es esencial subrayar que en el ordenador la ejecución del programa se efectúa *automáticamente*. El programa para el ordenador está codificado de modo especial y se registra de forma que sea accesible para la «percepción» de la máquina. En la mayoría de los casos la forma de registro más accesible para la máquina es la escritura en forma de agujeros perforados en la cinta (cinta perforada) o bien en tarjetas (tarjetas perforadas). Después de obtener la cinta perforada (o tarjetas perforadas) con el programa, el trabajo del ordenador se realiza bajo el control de esta cinta perforada (o tarjetas perforadas) sin intervención posterior del hombre. (Claro está que es fácil crear un dispositivo que asegure el funcionamiento automático, sin participación del hombre, de la máquina de Post.) En la primera aproximación el trabajo del ordenador puede ser comparado con el funcionamiento de un piano mecánico, en el cual la sucesión con que se aprietan las teclas está determinada exactamente por el programa registrado en la cinta perforada y se realiza sin participación del hombre.

Observación. Sin embargo, la analogía entre el ordenador y el piano mecánico tiene lugar sólo en la primera proximación. En la realidad, el funcionamiento del piano mecánico, en esencia, muy poco difiere del trabajo de un organillo. En éste los salientes y huecos del cilindro, durante su rotación, abren y cierran los canales de los tubos sonoros; cada semejante abertura (es decir, la abertura del canal de un tubo dado en un momento dado) se maneja mediante su propia desigualdad en la superficie del cilindro; el número de sones en una melodía es igual al número de estas desigualdades cuanto más larga es la melodía, tanto mayor deberá ser el cilindro. En el piano mecánico cada apriete de la tecla se suscita por cierto sector de la cinta perforada y cada semejante sector provoca exactamente un apriete tan sólo de una tecla; por consiguiente, cuanto más larga sea la melodía, tanto más largo deberá ser el programa registrado en la cinta perforada. Tanto en el organillo, como en el piano mecánico la escritura en el cilindro (mediante salientes y huecos en su superficie) y en la cinta perforada (mediante las perforaciones) es, en su género, una escritura de caracteres musicales. En el ordenador el asunto es otro. Aquí, como en la máquina de Post, una misma instrucción del programa puede ejecutarse muchas veces. Por esta razón, el número de pasos de funcionamiento de la máquina no está relacionado con la longitud del programa: un mismo programa puede asegurar cualquier número de pasos necesario para alcanzar el resultado. (Por ejemplo, el programa indicado por nosotros en calidad de solución del problema 2 del capítulo «Adición de la unidad en la máquina de Post» tiene la longitud 4, mientras que el número necesario de pasos para obtener el resultado, que se exige en este problema, depende de donde se encuentra el carro, al comenzar.) Esta es una propiedad importantísima y presta la posibilidad de realizar en la máquina los algoritmos y, precisamente, componer programas únicos que llevan al resultado para una clase entera de posibles datos iniciales. Esta propiedad se garantiza por la posibilidad de pasar del cumplimiento de una instrucción a la ejecución de cualquiera otra, mientras que en el piano mecánico o en el organillo después de cierta «instrucción» de apriete de la tecla o de abertura de los canales sonoros puede ejecutarse sólo la «instrucción» que la sigue directamente en el registro del cilindro o bien de la cinta perforada.

3. La distinción principal, más esencial entre el ordenador y la máquina de Post consiste en lo siguiente: en el memorizador (en la cinta de la máquina de Post) antes de comenzar el trabajo se registra el dato inicial (por ejemplo, el cortejo de números que están destinados a la adición). El programa, que debe ser aplicado a este dato inicial y de acuerdo con el cual trabaja la máquina, como que estuviera aparte. En el ordenador el programa, antes de comenzar el funcionamiento de la máquina, se introduce en el dispositivo de memoria (memorizador) junto con el dato inicial. (En esto, a propósito, consiste una distinción más entre el ordenador y el piano mecánico: en el último el programa registrado en la cinta perforada «se lee» por un dispositivo especial y a medida de la lectura transcurre la interpretación; por lo tanto, este dispositivo ejecuta el papel del pianista que toca por notas expuestas ante él. En el ordenador la cinta perforada con el programa y dato inicial registrados en ésta antes de empezar el trabajo de la máquina se procesa por un «dispositivo de entrada» especial, que lee la información que contiene la cinta perforada y coloca esta información en el dispositivo de memoria; en este aspecto la máquina se parece al pianista, que primeramente aprende las notas al tacto y después toca de memoria, sin mirar las notas.) El contenido del dispositivo de memoria en la máquina en cierto instante antes de empezar el trabajo, lo denominaremos información inicial. En la máquina de Post esta información consta sólo del dato inicial. En el ordenador la información inicial consta del dicho dato y del programa. Cabe subrayar, que la división de la información en la inicial y el programa es bastante convencional. Toda esta información, incluida aquella parte que se denomina «programa», puede ser sometida a varias transformaciones en el proceso de funcionamiento de la máquina. Esta posibilidad de cambiar las mismas instrucciones en el proceso de trabajo, que no hay en la máquina de Post, es un rasgo muy importante de los ordenadores modernos. Por otra parte, el contenido de uno u otro sector del dispositivo de memoria se puede, con frecuencia, interpretar como instrucción. Por este motivo, la división de la información contenida en el dispositivo de memoria en el programa y dato inicial, en general, tiene sentido sólo en el primer instante, luego, en cada paso toda esta información, como un todo, se somete al procesamiento.

Es natural el interrogente: ¿según qué ley transcurre el procesamiento de la información contenida en el dispositivo de memoria? Este procesamiento transcurre, a su vez, a base de cierta regla que llamaremos Programa Universal. El Programa Universal de ninguna manera debe ser confundido con el programa particular destinado para resolver uno u otro problema específico y que aplica a los datos iniciales. El programa particular depende del problema a resolver, mientras que el Programa Universal es uno para todos los problemas; en los ordenadores modernos se introduce en la propia estructura de la máquina (por esta razón, estas máquinas, a menudo, se llaman *universales*). Hagamos el balance de lo dicho. En la máquina de Post al comienzo se introduce el dato inicial, después de lo que el trabajo transcurre según el programa que no se encuentra en el dispositivo de memoria. En el ordenador primeramente en el dispositivo de memoria se coloca la información inicial constituida por el programa particular para resolver el problema prefijado y por el dato inicial, después de lo que el funcionamiento (precisamente el procesamiento de todo el contenido del dispositivo de memoria) transcurre según el Programa Universal encarnado en la estructura de la máquina[1]).

Puede ser indicada una distinción más entre el ordenador y la máquina de Post, que parecería muy esencial: el memorizador de la máquina de Post —la cinta— tiene una capacidad infinita, lo que no puede ocurrir en las máquinas reales. Sin embargo, también en la máquina de Post la cinta infinita puede ser sustituida por una finita a la que se pega otra cuando esto es necesario (pues, a pesar de todo, para cada momento de funcionamiento de la máquina de Post será utilizado un pedazo finito de la cinta). Por otra parte, el dispositivo de memoria externa del ordenador también puede, en principio, aumentar indefinidamente su capacidad, añadiendo nuevas partes (digamos, nuevas cintas magnéticas). Así pues, tanto la máquina de Post, como el ordenador pueden considerarse poseedores de un memorizador que es finito en cada momento dado, pero de capacidad creciente indefinidamente. Precisamente esta circunstancia, emparienta, de modo principal, la máquina de Post con el ordenador y permite examinarla como modelo simplificado de éste.

---

[1]) Por lo demás, se puede componer el Programa Universal también para la máquina de Post, precisamente, puede ser codificado cada programa para la máquina de Post en forma de la palabra de Post y ubicar, a continuación, el registro de este programa (es decir, la escritura de la palabra correspondiente) en la cinta junto al registro de aquel dato inicial al cual dicho programa debe aplicarse. Después se puede elaborar el Programa Universal que siendo aplicado al registro compuesto de la escritura de cierto programa P y del registro de cierto dato inicial x diera el mismo resultado que la aplicación directa del P al registro x. Sin embargo, en este caso el Programa Universal será sólo uno de los posibles programas para la máquina de Post: pero, en el caso del ordenador el Programa Universal no es, en absoluto, uno de los programas particulares tolerables para el ordenador, sino que está realizado por la misma estructura del ordenador.

En calidad de suplemento proponemos una traducción del inglés al español del célebre artículo de Emil L. Post «Procesos combinatorios finitos, enunciación 1» (Emil L. *Post* «Finite combinatory processes—formulation 1»). Este artículo fue publicado en el número 3, correspondiente a septiembre de 1936, del 1-er volumen de la «Revista de la lógica simbólica» («The Journal of Symbolic Logic») que aparece cada trimestre. La publicación fue acompañada por la siguiente nota del editor: «Recibido el 7 de octubre de 1936. Aconsejamos al lector que compare el artículo de A. M. Turing «Acerca de los números computables» («On computable numbers») que deberá salir a luz proximamente en las «Obras de la sociedad matemática de Londres» («Proceedings of the London Mathematical Society»). Sin embargo, a pesar de que el presente artículo tiene una fecha más tardía, está escrito independientemente, en absoluto, del artículo de A. M. Turing». El artículo de A. M. Turing (que tiene fecha de entrada en la redacción de la revista el 28 de mayo de 1936) fue publicado en aquel mismo año 1936. (A. M. Turing, On computable numbers, with an application to the Entscheidungsproblem.—Proceedings of the London Mathematical Society, ser. 2, 1936, 42, N° 3—4, p. 230—265. A correction.—Idem, 1937, 43, N° 7, p. 544—546).

Así pues,

PROCESOS COMBINATORIOS FINITOS, ENUNCIACIÓN 1*)

Por Emil L. Post

La presente enunciación puede tener importancia significativa para el desarrollo de la lógica simbólica según el planteamiento del teorema de Gödel sobre la incompletitud de ló-

) Traducción del texto inglés publicado en la «Revista de la lógica simbólica» («The Journal of Simbolic Logic»), vol. 1, № 3, septiembre 1936.—(Nota del traductor.)*

gicas simbólicas[1]) y del resultado de Church en lo que concierne a los problemas insolubles absolutamente[2]).

Tenemos en cuenta el *problema general* que consta de una serie de *problemas específicos*. La solución del problema general será aquella que dé respuesta a cada uno de los problemas específicos.

En la enunciación expuesta a continuación de lo que es solución se hace uso de dos conceptos: el *espacio de símbolos*, en el cual debe realizarse el trabajo que lleva del problema a la respuesta[3]) y el *juego de instrucciones* inalterado fijado que gobernará las operaciones en el espacio de símbolos y determinará el orden, en el cual estas instrucciones deben ser aplicadas.

En la presente enunciación el espacio de símbolos consta de una secuencia infinita por ambos lados de espacios o cajas*); por lo tanto desde el punto de vista del orden, aquél es similar a la serie de números enteros ..., —3, —2, —1, 0, 1, 2, 3. ,... El «solucionador» del problema o bien el operario puede moverse y trabajar en este espacio de símbolos, siendo capaz de estar y obrar en cada uno de los momentos en una sola caja. Aparte de la presencia del operario, cada caja puede admitir un solo estado de los dos posibles, a saber: estar vacía o no marcada, o bien tener una marca única, digamos, una raya vertical.

Una caja debe ser destacada, la llamaremos punto de partida. A continuación, supongamos que el problema específico debe prefijarse en forma simbólica mediante un número finito de cajas marcadas con una raya. Pues, la respuesta, de modo análogo, se presenta en forma simbólica mediante una configuración similar de cajas marcadas. De manera más concreta la respuesta representa de por sí la configuración de las

---

1) Kurt *Gödel*, Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I, Monatshefte für Mathematik und Physik, 1931, vol. 38, N° 1, p.p. 173...198. (Todas las notas del artículo de E. L. Post aquí y en adelante pertenecen a su autor.—*Nota del traductor*.)

2) Alonzo *Church*, An unsolvable problem of elementary number theory, American Journal of Mathematics, 1936, vol. 58, N°2, p.p. 345...363.

3) Espacio de símbolos y tiempo.

*) En el presente libro esto se refiere a las células.—(*Nota del traductor*.)

cajas marcadas que aparece una vez concluido el proceso de solución.

Se supone que el operario es capaz de efectuar las siguientes operaciones primitivas:[4])

(a) marcar la caja, en la cual él se encuentra (si ésta está vacía);

(b) borrar la marca en la caja, en la cual él se encuentra (si ésta está marcada);

(c) pasar a la caja que está a la derecha de él;

(d) pasar a la caja que está a la izquierda de él;

(e) determinar si está marcada o no la caja, en la cual él se encuentra.

El juego de las instrucciones que, cabe subrayar, es el mismo para todos los problemas específicos y, por lo tanto, corresponde al problema general, debe tener la forma siguiente. Dicho juego comienza así:

Márchense del punto de partida y sigan la instrucción 1.
El juego contiene un número finito de instrucciones numeradas por, 1, 2, 3, . . ., n. La i-ésima instrucción debe tener uno de los siguientes aspectos:

(A) *ejecuten la operación* O_i [O_i) = (a), (b), (c) o (d)] *y, luego, sigan la instrucción* j_i;

(B) *ejecuten la operación* (e) *y, de acuerdo con la respuesta si o no, sigan la instrucción* j_i' *o bien* j_i'';

(C) *Stop.*
Claro está, que se requiere una sola instrucción de tipo (C). Además, notemos que el estado del espacio de símbolos influye directamente sobre el proceso sólo por medio de las instrucciones de tipo (B).

Diremos, que el juego de instrucciones es *aplicable* al problema general dado si su aplicación a cada problema específico nunca exigirá la operación (a), cuando la caja en la cual se encuentra el operario está marcada, o bien la operación (b), cuando la caja no está marcada [5]). El juego de instrucciones aplicable al problema general, siendo aplicado a cada problema específico, prefija el proceso determinante. Este pro-

[4]) Es lo mismo que seguir las instrucciones expuestas a continuación.

[5]) Aunque nuestra enunciación del juego de instrucciones puede ser variada de manera que la aplicación será garantizada inmediatamente, esto nos parece indeseable por varias razones.

ceso terminará cuando y sólo cuando se obtenga una instrucción de tipo (C). Diremos, que el juego de instrucciones prefija el *1-proceso finito* respecto al problema general, si aquél es aplicable a este problema y si el proceso *determinado por el juego culmina para cada problema específico.* El 1-proceso finito asociado con el problema general lo llamaremos *1-solución* de este problema, si su respuesta para cada problema específico siempre es correcta.

Aquí no nos ocupamos de qué modo la configuración de las cajas marcadas, que corresponde al problema específico y a su respuesta, simboliza el problema y la respuesta sensatos. En efecto, todo lo dicho antes supone que el problema específico se prefija en forma simbolizada por cierta fuerza exterior y la respuesta simbólica se percibe de manera análoga. El perfeccionamiento que hace los planteamientos de mayor autonomía es el siguiente. Claro está, que el problema general consta no más que de en un conjunto numerable infinito de problemas específicos. No consideraremos el caso finito. Sea que entre la clase de números enteros positivos y la de problemas específicos se ha establecido una correspondencia biunívoca. Nosotros podemos, de modo suficientemente arbitrario, representar el entero positivo *n*, al marcar *n* en las primeras cajas a la derecha del punto de partida. Denominaremos el problema general *1-prefijado*, si ha sido constituido un 1-proceso finito tal, que siendo aplicado a la clase de números enteros positivos, representados simbólicamente, según lo recién indicado, presta, de modo biunívoco, la clase de problemas específicos, que forma el problema general. Es conveniente para lo posterior considerar que, cuando el problema general es 1-prefijado del modo indicado, cada proceso específico durante su terminación deja al operario en el punto de partida. Si, ahora, cierto problema general es 1-prefijado y 1-soluble, podemos con evidentes cambios combinar los dos juegos de instrucciones para obtener el 1-proceso finito, que da respuesta a cada problema específico, cuando el último está prefijado simplemente por su número en forma simbólica.

Con cierta modificación la enunciación expuesta también es aplicable a las lógicas simbólicas. Ahora debemos tratar no la clase de problemas específicos, sino una sola marcación inicial finita del espacio de símbolos, la cual simboliza las afirmaciones primitivas formales de la lógica. Por otro lado,

ahora no tendremos instrucciones de tipo (C). Por consiguiente, al suponer la posibilidad de aplicación, se prefija el proceso determinante que es *infinito.* A continuación suponemos, que durante este proceso aparecen ciertos grupos de símbolos que pueden ser reconocidos, es decir, secuencias finitas de las cajas marcadas y no marcadas, las cuales en adelante durante el proceso no se alteran. Éstas representarán de por sí afirmaciones deducidas de la lógica. Está claro, que el juego de instrucciones corresponde a los procesos deductivos de la lógica. En este caso la lógica puede llamarse *1-generable.*

Sin embargo, el procedimiento alternativo de menor concordancia con el espíritu de la lógica simbólica podría consistir en la indicación de 1-proceso finito, que presta el *n*-ésimo teorema o bien la *n*-ésima afirmación formal de la lógica, en vista de que de nuevo está presentado *n* simbolizado, como antes.

Nuestra idea inicial acerca del problema específico prefijado lleva a una dificultad, la cual vale la pena de recordar. A saber, si una fuerza exterior prefija la marcación inicial finita del espacio de símbolos, nosotros no tenemos procedimiento para determinar, por ejemplo, cuál de las cajas marcadas es la primera y cuál, la última. Esta dificultad se evita completamente, cuando el problema general es el 1-prefijado. Dicha dificultad también resulta eludida con éxito cada vez, en que se indica el 1-proceso. En la práctica, los problemas específicos entendidos pueden ser simbolizados de tal modo, que los límites de semejante simbolización se reconocen mediante los grupos característicos de cajas marcadas y no marcadas.

Sin embargo, la raíz de nuestra dificultad se encuentra, con probabilidad, en nuestra admisión del espacio de símbolos infinito. En la presente enunciación las cajas son, por lo menos en sentido especulativo, entes físicos, por ejemplo, las células contiguas. Nuestra fuerza exterior puede presentarnos un número infinito de tales cajas no en mayor grado que marcar un número infinito de las que ya están presentadas. Pero, si esta fuerza nos presenta el problema específico, como último pedazo del espacio de símbolos, la dificultad examinada desaparece. Es lógico, que esto requeriría extender la relación de las operaciones primitivas, para que fuera permitido el aumento necesario del espacio de símbolos prefijado infinito,

a medida del desarrollo del proceso. Por consiguiente, la versión final de la enunciación del presente tipo también debería contener instrucciones para la generabilidad del espacio de símbolos[6]).

El autor abriga esperanzas que la presente enunciación resulte lógicamente equivalente a la recursividad en el sentido de la deducción de Gödel—Church[7]). Sin embargo, el objetivo de la enunciación consiste en proponer un sistema no sólo de cierta fuerza lógica, sino que también, en sentido restringido, de fidelidad psicológica. En este último sentido deben ser examinadas las enunciaciones cada vez más y más amplias. Por otro lado, nuestro objetivo será mostrar que todas éstas lógicamente son reducibles a la enunciación 1. En el presente momento nosotros proponemos esta conclusión como *hipótesis de trabajo*. Según nuestra opinión es semejante también la identificación, propuesta por Church, de calculabilidad efectiva con recursividad [8]). De esta hipótesis, y a causa de la contradicción manifiesta con todo el desarrollo matemá-

[6]) En sus estadios iniciales el desarrollo de la enunciación 1 tiende a ser un poco complicado. Aunque esto discuerda con el espíritu de semejantes enunciaciones, su forma definitiva puede sacrificar su presente simplicidad a favor de mayor flexibilidad. Una de las posibilidades es la de tener más procedimientos que uno para marcar la caja. No se excluye que la naturalidad deseable puede ser alcanzada por medio de la tolerancia del número finito, puede ser de dos objetos físicos, destinados para utilizarlos como indicadores, con que el operario podrá identificarlos y trasladarlos de una caja a la otra.

[7]) Es posible que la comparación requerida sería más fácil de efectuar al determinar el concepto de 1-función y al demostrar la equivalencia de esta definición a la de la función recursiva. (Véase la obra de Church, p. 350, indicada en la nota[2]) del presente artículo.) Con ello, por la 1-función se entendería tal función $f(n)$ con argumentos y valores positivos enteros, para la cual puede ser indicado el 1-proceso finito con la siguiente propiedad: para cada n positivo entero, tomado como problema, el proceso da $f(n)$ en calidad de respuesta, con la particularidad de que n y $f(n)$ se simbolizan como lo concordamos antes.

[8]) Compárese la obra de Church, p.p. 346, 356... 358, indicada en la nota[2]) del presente artículo. Por lo demás, el trabajo realizado por Church y otros lleva esta identificación muy lejos fuera de los estadios de la hipótesis de trabajo. Pero, el intento de enmascarar la identificación por la definición oculta el hecho de que está efectuado un descubrimiento que se refiere a los límites de la fuerza matematizada de *Homo Sapiens* y nos hace ciegos con respecto a la necesidad de verificar continuamente esta identificación.

tico, a partir de la demostración de Cantor de la innumerabilidad de los puntos en una resta, se desprenden independientemente los resultados de Gödel—Church. El éxito del programa expuesto antes consistiría, para nosotros, en la conversión de esta hipótesis no tanto en una definición o bien en un axioma, como en una ley de la naturaleza. Sólo así, considera el autor, el teorema de Gödel sobre la incompletitud de lógicas simbólicas de cierto tipo general y el resultado de Church de la insolubilidad recursiva de algunos problemas, pueden ser transformados en conclusiones concernientes a todas las lógicas simbólicas y a todos los métodos de solución.

Colegio de la ciudad de Nueva York
(College of the City of New York)

Lecciones populares de matemáticas

Obras de nuestro sello editorial

V. Uspenski.
Teorema de Gödel sobre la incompletitud

A. Smogorzhevski.
Acerca de la geometría de Lobachevski

A. Markushévich.
Curvas maravillosas

S. Shílov.
Análisis matemático en el campo de funciones racionales

Editorial MIR



Moscú